Oracle® Fusion Middleware

Developer's Guide for Oracle WebCenter Portal
(Oracle Fusion Applications Edition)

11*g* Release 1 (11.1.1.6.2)

E25595-01

April 2012

ORACLE®

Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal (Oracle Fusion Applications Edition), 11*g* Release 1 (11.1.1.6.2)

E25595-01

Copyright © 2007, 2012, Oracle and/or its affiliates. All rights reserved.

Primary Author: William Witman

Contributing Authors: Ingrid Snedecor, Jennifer Shipman, Joan Carter, Lalithashree Rajesh, Michele Cyran, Peter Jacobsen, Promila Chitkara, Rosie Harvey, Sarah Bernau, Savita Thakur, Sue Highmoor, Tom Maremaa

Contributor: Randy Akl, Maheswaran Anantharaman, Sivakumar Balagopalan, Rahmathulla Baig, Ravishankar Belavadi, Chris Broadbent, Steve Burns, Chris Carter, Chung Cheng, Demetris Christou, Vicki Chun, Ross Clewley, Manish Devgan, Marcus Diaz, Sumit Dubey, Frans Effendi, Paul Encarnacion, Gaurav Mittal, Jeni Ferns, Robin Fisher, Michael Freedman, Nick Greenhalgh, Christian Hauser, Harsha Huddar, Hsing Huang, Clayton Jung, Medini Kakade, Hrishikesh Karambelkar, Seshan Kennan, Sanjay Khanna, Vasant Kumar, Paul Lin, Yueh-Hong Lin, Alison Macmillan, George Maggessy, Pankaj Mittal, Dan Mullen, Lei Oh, Nicolas Pombourcq, Prabhakar Munnangi Reddy, Puvanenthiran Subbaraj, Rajesh Ramachandran, Sripathy Rao, Shubha Rangarajan, Shakeb Sagheer, Raghu Sampathkrishna, Andrew Sefkow, Jennifer Shu, Ved Singh, Vidhya Sundaram, Paul Spencer, Stephen Thornhill, Deepthi Umakanth, Kundan Vyas, Alistair Wilson, Stewart Wilson, Hui Zeng, Brad Posner, Steve Roth, Greg Smith, Tanya Saarva, Chris Bales, Ray Polk, Jeff Mueller, Pankash Kapasi, Sameer Sawant, Ken Young, Leevar Williams, Arun Tharakkal, Don Hayler, Cindy McMullen, Phil Griffin, Tom Porcaro, Dawn Tyler, Jordan Douglas

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface lxi

Audience lxi
Documentation Accessibility lxii
Related Documents lxii
Conventions lxii

What's New lxiii

New Features for Release 11.1.1.6 lxiii

Part I Introduction

1 Developer's Quick Start Guide

1.1 WebCenter Portal Developer's FAQ 1-1
1.2 Installing Required Software 1-3
1.3 Preparing for Team Development 1-3
1.4 Creating a New Framework Application 1-3
1.5 Preparing for Iterative Development 1-4
1.5.1 What Is Iterative Development? 1-4
1.5.2 How Does Iterative Development Work? 1-4
1.5.3 Enabling Iterative Development 1-6
1.5.4 Turning Off Iterative Development 1-6
1.6 Creating Application Resource Connections 1-7
1.6.1 Where Are Connections Located? 1-7
1.6.2 How Do I Access the Connection Wizards? 1-7
1.6.3 Where Can I Learn More About Connections? 1-8
1.7 Planning the Portal Layout 1-8
1.8 Working with Skins 1-8
1.9 Preparing for Round-Trip Development 1-8
1.10 What Configuration Files Should I Know About? 1-9
1.10.1 The Framework Application Template Default web.xml File 1-9
1.10.2 The Framework Application Template Default faces-config.xml File 1-10
1.10.3 The Framework Application Template Default trinidad-config.xml File 1-11
1.10.4 The Framework Application Template Default adfc-config.xml File 1-12
1.11 Installing Database Schemas 1-12

2 Understanding Oracle WebCenter Portal

2.1 What Is Oracle WebCenter Portal? ... 2-1
2.2 Introducing the WebCenter Portal: Framework ... 2-2
2.3 What Are the Main WebCenter Portal: Framework Features? ... 2-2
2.3.1 Portlets and Pagelets for Accessing Remote Applications and Services ... 2-2
2.3.1.1 What Are Portlets? ... 2-3
2.3.1.2 What Are Pagelets? ... 2-3
2.3.1.3 What is the Difference Between Portlets and Task Flows? ... 2-3
2.3.2 Oracle WebCenter Portal's Composer Allows User Customization ... 2-4
2.3.3 Content Integration with Supported Content Repositories ... 2-4
2.3.4 Search Framework ... 2-5
2.3.5 Resource Catalogs Help Organize the Development Experience ... 2-5
2.3.6 Application Security ... 2-6
2.4 Introducing Oracle WebCenter Portal: Services ... 2-7
2.4.1 WebCenter Portal: Services and the REST APIs ... 2-9
2.5 Introducing Oracle Application Development Framework (Oracle ADF) ... 2-10
2.5.1 What Is Oracle ADF? ... 2-10
2.5.2 Why Is Oracle ADF Important for WebCenter Portal Developers? ... 2-10
2.5.3 What Does Oracle ADF Technology Include? ... 2-10
2.6 Introducing Oracle JDeveloper and the WebCenter Portal Extension ... 2-10
2.7 Understanding the Application Life Cycle ... 2-12
2.8 Design Questions to Consider Before You Start ... 2-12
2.8.1 User Considerations ... 2-12
2.8.2 Site Administrator Considerations ... 2-14
2.8.3 Developer Considerations ... 2-14
2.9 Overview of WebCenter Accessibility Features ... 2-16
2.9.1 Generating Accessible HTML ... 2-16
2.9.2 Accessibility Features at Runtime ... 2-17
2.9.3 Accessibility Considerations for Portlets ... 2-17

3 Preparing Your Development Environment

3.1 Installing Oracle JDeveloper ... 3-1
3.2 Installing WebCenter Portal's Extension for Oracle JDeveloper ... 3-1
3.3 Setting the User Home Directory Environment Variable ... 3-2
3.4 Working with the Integrated WebLogic Server ... 3-2
3.4.1 How to Start and Stop Integrated WLS ... 3-2
3.4.2 What You May Need to Know About Integrated WLS ... 3-3
3.4.3 The WebCenter Preconfigured Server Readme File ... 3-3
3.4.4 What You May Need to Know About Preconfigured Portlet Producers ... 3-4
3.4.4.1 OmniPortlet and Web Clipping ... 3-4
3.4.4.2 The WSRP Sample Portlet Producers and Portlets ... 3-5
3.4.4.3 The PDK-Java Sample Portlet Producer and Portlets ... 3-6
3.5 Installing Database Schemas ... 3-6
3.5.1 Installing Schemas With RCU ... 3-7
3.5.2 Installing WebCenter Portal's Schema with a SQL Script ... 3-7
3.6 Installing the Oracle WebCenter Adapter for SharePoint ... 3-7

4 Working Productively in Teams

4.1 Enabling Source Control on Framework Applications 4-1
4.1.1 Creating a Subversion Repository 4-2
4.1.2 Understanding Oracle Team Productivity Center 4-3
4.2 Understanding Framework Application Files Affected by Developers 4-4
4.2.1 Files Associated With Common Objects 4-4
4.2.2 Developer Actions Affecting Metadata Files 4-4
4.3 Implementing Common Requirements One Time 4-12
4.4 Portlet Producer Considerations 4-13
4.4.1 Portlet Producer Connections 4-13
4.4.2 Portlet Producer Name Clashes 4-13
4.4.3 Combining Portlets from Different Portlet Producers 4-13

5 Creating Framework and Portlet Producer Applications

5.1 What Are Application Templates? 5-1
5.2 Creating a Framework Application 5-2
5.2.1 How to Create a Framework Application 5-3
5.3 Adding Pages to a Portal 5-8
5.4 Understanding the PortalWebAssets Project 5-9
5.4.1 Introduction 5-9
5.4.2 Decoupling the Static Application Resources Project 5-9
5.4.3 Deployment Options 5-11
5.4.4 Using EL to Manage Static Resources 5-11
5.4.4.1 Defining a Base URL Preference 5-11
5.4.4.2 Mapping URLs Dynamically 5-11
5.5 Creating a Portlet Producer Application 5-12
5.5.1 How to Create a Portlet Application Using a Template 5-13
5.5.2 What Happens When You Use a Portlet Producer Application Template 5-13
5.6 How Is a Framework application Security Configured by Default? 5-14

6 Advanced Configuration Options

6.1 Manually Configuring a Framework Application With Technology Scopes 6-1
6.1.1 What are Technology Scopes? 6-1
6.1.2 Adding and Removing Technology Scopes 6-2
6.1.3 What Are the Default Technology Scopes for a Framework Application? 6-4
6.1.4 Which Technology Scopes Are WebCenter Portal: Framework Application-Specific? ... 6-5
6.1.5 Can I Create a WebCenter Portal: Framework Application by Adding the Oracle WebCenter Portal: Framework Technology Scope Later? 6-5
6.1.6 Migrating a WebCenter PS2 or Earlier Application to a WebCenter Portal: Framework Application 6-5
6.2 Extending Non-WebCenter Portal: Framework Applications to Include WebCenter Capabilities 6-5
6.3 Creating a Project by Importing a WAR File 6-6

7 Preparing Your Application for WebCenter Portal Services

7.1 Understanding WebCenter Portal Services Technologies 7-1
7.1.1 Understanding WebCenter Portal Horizontal Services 7-1
7.1.2 Understanding WebCenter Portal APIs 7-2
7.1.3 Using WebCenter Portal Data Controls 7-5
7.2 Preparing Your Framework Application to Consume Services 7-6
7.2.1 How to Prepare Your Application to Consume Services 7-6
7.2.1.1 Implementing Security for Services 7-7
7.2.1.2 Setting Up SSL-Protected Connections for Services 7-8
7.2.2 Setting Up a Database Connection 7-9
7.2.2.1 Associating Existing Database Connections 7-12
7.2.3 Setting Up an External Application Connection 7-12
7.3 Extending Your Framework Application with Custom Components 7-13
7.3.1 Introducing the Resource Action Handling Framework 7-13
7.3.2 Registering a Resource Viewer 7-14
7.3.3 Creating a Resource Authorizer Class 7-18
7.3.4 Specifying the View Handler and Navigation Behavior 7-19
7.4 Configuring General Settings for Your Services 7-21
7.4.1 Building a Preferences User Interface 7-23
7.4.2 Using General Settings to Specify the User's Locale 7-24

8 Introducing the AviTrust Portal Framework Sample Application

8.1 About the AviTrust Portal Framework Application for WebCenter 8-1
8.2 Downloading the AviTrust Portal Framework Application 8-2
8.3 Browsing Portal Framework Features at Runtime 8-2
8.3.1 Exploring the AviTrust Sample Portal Home Page 8-3
8.3.2 Navigating the Site Using Tabs 8-6
8.4 Accessing and Using the Resource Manager 8-11
8.5 In-Context Editing of Web Content at Runtime 8-15
8.6 Exposing Social Computing Services 8-19
8.6.1 Adding a Discussion Forum 8-20
8.7 WebCenter Framework Components in the AviTrust Portal Demo 8-21

Part II Setting Up Your Portal Structure

9 Understanding Framework Applications

9.1 What is a Framework Application? 9-1
9.2 What is Oracle WebCenter Portal: Framework? 9-3
9.2.1 How Do I Add Oracle WebCenter Portal: Framework to a Project? 9-3
9.2.2 What Features Does the Oracle WebCenter Portal: Framework Provide? 9-3
9.3 What is the Portal Life Cycle? 9-4
9.4 Creating a WebCenter Portal: Framework Application 9-4
9.5 Understanding the Features of a WebCenter Portal: Framework Application 9-5
9.5.1 Understanding Pages, Page Templates, and the Portal Page Hierarchy 9-5
9.5.2 Securing Your Portal Pages 9-9
9.5.3 Understanding the Navigation Model and the Navigation Registry 9-11

9.5.4 Understanding Resource Catalogs and the Catalog Registry 9-13
9.5.5 Other Design Time Folders 9-15
9.5.6 Tips for More Advanced Development Use Cases 9-15
9.6 Developing Your Portal's Look and Feel 9-16
9.7 Changing Default Portal Preferences 9-16
9.7.1 What Are the Default Portal Preferences? 9-16
9.7.2 How to Change the Default Preferences at Design Time 9-17
9.8 Using Iterative and Round-Trip Development Techniques 9-17
9.9 What is the Application Sources Folder? 9-18
9.10 Running and Testing a Portal 9-18
9.11 WebCenter Portals at Runtime 9-19
9.11.1 Preserving Runtime Customizations on the Integrated WebLogic Server 9-19
9.11.2 Runtime WebCenter Portal Administration Console 9-20
9.11.3 How Security Settings Affect the Runtime Portal? 9-21
9.11.4 Runtime Editing of Portal Resources 9-21
9.12 How are WebCenter Portal Files Organized? 9-22
9.12.1 Understanding the Organization of a WebCenter Portal: Framework Application . 9-23
9.12.1.1 How is a WebCenter Portal: Framework Application Organized? 9-23
9.12.1.2 Why Is a WebCenter Portal: Framework Application Organized the Way It Is? 9-24
9.12.2 Viewing Your Portal Project on the Filesystem 9-24
9.12.3 Viewing Your Portal Project in JDeveloper 9-25
9.13 Changing the Default Home Page and Login/Logout Target Pages 9-26
9.13.1 Understanding How the Home Page is Specified 9-26
9.13.2 How to Change the Default Home Page 9-27
9.13.3 How is the Default Index Page Specified? 9-28
9.13.4 Specifying the Target Page After a Login or Logout 9-29
9.14 Basic Portal Development Tasks 9-30

10 Understanding the WebCenter Portal Life Cycle

10.1 What is the WebCenter Portal Life Cycle? 10-1
10.2 What Are the Major Life Cycle Tasks? 10-3
10.2.1 One-Time Setup Tasks 10-4
10.2.2 Development Environment Tasks 10-4
10.2.3 Nightly Build Environment Tasks 10-5
10.2.4 Testing Environment Tasks 10-5
10.2.5 Stage Environment Tasks 10-5
10.2.6 Production Environment Tasks 10-5
10.3 Who Participates in the Portal Life Cycle? 10-6
10.4 Understanding the Build and Test Environments 10-6
10.5 Understanding the Staging and Production Environments 10-8
10.5.1 Provisioning the Staging Environment 10-9
10.5.2 Adding Content to the Staging Environment 10-9
10.5.3 Moving the Portal from Staging to Production 10-9
10.6 Tools for Managing the Life Cycle 10-10
10.7 Configuring a Nightly Build Script 10-10
10.8 Setting Up a Staging or Production Environment for the First Time 10-11

10.9 Moving a WebCenter Portal: Framework Application to an Existing Environment 10-11
10.10 Deploying to Managed Servers 10-12
10.11 Deploying and Configuring the Application on Targeted Servers 10-12
10.11.1 Introduction 10-12
10.11.2 Using the Deploy and Configure Script 10-12
10.11.3 Managing Post-Deployment Changes 10-15
10.12 Building and Deploying Personalization for WebCenter Portal Files 10-15
10.12.1 Building the WCPS MAR File 10-15
10.12.2 Deploying WCPS Files to a Server 10-17
10.12.3 Deploying WCPS Files to an Archive File 10-17
10.12.4 Importing and Exporting MAR Files 10-18
10.13 Using the Propagation Tool to Propagate From Staging to Production 10-18
10.13.1 Introduction to the Propagation Tool 10-19
10.13.2 Configuring the Propagation Tool 10-20
10.13.2.1 Configuring the URL Connection With Oracle Enterprise Manager Fusion Middleware Control Console 10-20
10.13.2.2 Configuring the URL Connection with WLST 10-21
10.13.3 Propagating Portal Metadata 10-22
10.13.4 Propagating WCPS Files 10-22
10.14 Propagating Content From Oracle WebCenter Content 10-23
10.15 Managing Security Through the Life Cycle 10-23
10.15.1 Migrating Security Policies for First-Time Deployment 10-23
10.15.2 Migrating Security Policies On Redeployment 10-24
10.15.3 Migrating Credentials for First-Time Deployment 10-24
10.15.4 Migrating Credentials On Redeployment 10-24
10.16 Migrating Portlet Preferences 10-25
10.17 Rolling Back Production Site Changes 10-25

11 Building a Navigation Model for Your Portal

11.1 Introduction to Navigation Models 11-1
11.2 Creating a Navigation Model 11-2
11.3 Adding Resources to a Navigation Model 11-4
11.3.1 How to Drag and Drop a Resource onto a Navigation Model 11-5
11.3.2 How to Add a Content Item to a Navigation Model 11-5
11.3.3 How to Add Content Based on the Results of a Query to a Navigation Model 11-7
11.3.4 How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model 11-9
11.3.5 How to Add a Page Hierarchy to a Navigation Model 11-13
11.3.6 How to Embed Another Navigation Model Within a Navigation Model 11-15
11.3.7 How to Extend Navigation Model Data 11-16
11.3.7.1 How to Add a Link to a Custom Resource 11-17
11.3.7.2 How to Add a Component to a Navigation Model 11-18
11.3.7.3 How to Add a Custom Folder to a Navigation Model 11-20
11.3.7.4 How to Add a Custom Content Provider to a Navigation Model 11-21
11.3.8 How to Organize Resources in a Navigation Model 11-23
11.3.9 How to Set Display Options for a Resource 11-23
11.4 Selecting the Default Navigation Model 11-25

11.5 Editing the Navigation Renderer 11-26
11.6 Setting the REDIRECT_OPTIONS Context Parameter 11-27
11.7 Enabling Runtime Administration of Navigation Models 11-27
11.7.1 How to Edit the Navigation Registry File 11-28
11.7.2 How to Enable Further Development of Deployed Navigation Models 11-29
11.7.2.1 Importing a Navigation Model 11-29
11.7.2.2 Exporting a Navigation Model 11-29
11.8 Working with WebCenter Portal: Spaces Navigation Models 11-29

12 Designing the Look and Feel of Your Portal

12.1 Introduction to Page Templates and Styles 12-1
12.1.1 Page Templates 12-2
12.1.2 Page Styles 12-4
12.1.3 Mashup Styles 12-6
12.1.4 Content Presenter Display Templates 12-7
12.2 Working with Page Templates 12-7
12.2.1 Introduction to the WebCenter Portal Application Seeded Page Templates 12-8
12.2.2 How to Create a Page Template 12-9
12.2.3 What You Should Know About Creating Page Templates for WebCenter Portal Applications 12-14
12.2.4 Enabling Runtime Switching of Page Templates 12-15
12.2.5 How to Edit a Page Template 12-17
12.2.6 How to Delete a Page Template 12-17
12.2.7 How to Enable Runtime Administration of a Page Template 12-17
12.2.8 How to Enable Further Development of Deployed Page Templates 12-18
12.2.8.1 Importing a Page Template 12-18
12.2.8.2 Exporting a Page Template 12-18
12.2.9 How to Create and Manage WebCenter Portal: Spaces Page Templates 12-19
12.3 Working with Page Styles 12-19
12.3.1 How to Create a Page Style 12-19
12.3.2 How to Enable Runtime Administration of a Page Style 12-22
12.3.3 How to Enable Further Development of Deployed Page Styles 12-23
12.3.3.1 Importing a Page Style 12-23
12.3.3.2 Exporting a Page Style 12-24
12.3.4 How to Create and Manage Page Styles for Spaces 12-24
12.4 Working with Mashup Styles 12-25
12.4.1 How to Create a Mashup Style 12-26
12.4.2 How to Enable Runtime Administration of Mashup Styles 12-28
12.4.3 How to Enable Further Development of Mashup Styles in a Deployed Application 12-29
12.4.3.1 Importing Mashup Styles 12-29
12.4.3.2 Exporting Mashup Styles 12-30
12.4.4 How to Create and Manage Mashup Styles for Spaces 12-30

13 Adding Resources to Your Portal

13.1 Adding Pages to Your Portal 13-1

13.2 Adding Service Task Flows to Your Portal 13-2
13.3 Adding Custom Task Flows to Your Portal 13-3
13.4 Adding Content from Content Server to Your Portal 13-3
13.5 Adding Portlets to Your Portal 13-4
13.6 Adding Pagelets to Your Portal 13-4
13.7 Adding External Applications to Your Portal 13-5
13.8 Adding Data Controls to Your Portal 13-5

14 Visualizing Your Portal Navigation

14.1 Introduction to Navigation Visualization 14-1
14.2 Adding Built-in Navigation Task Flows to a Page Template 14-2
14.2.1 How to Add a Navigation Task Flow to a Page Template 14-3
14.2.2 Navigation Task Flow Parameters 14-4
14.2.3 What You May Need to Know About Navigation Task Flow Style Classes 14-9
14.3 Programmatically Adding Navigation to a Page Template 14-10
14.3.1 Using the Navigation Expression Language APIs 14-10
14.3.1.1 Introduction to the Navigation Context 14-11
14.3.1.2 Introduction to the Navigation Runtime Model 14-12
14.3.1.3 Introduction to the Navigation Resource 14-14
14.3.1.4 How to Render the Navigation Model as a List of Links 14-16
14.3.1.5 How to Render the Navigation Model as a Tree 14-17
14.3.1.6 How to Render the Navigation Model as a Three-Level Menu 14-17
14.3.1.7 How to Render the Navigation Model as Breadcrumbs 14-18
14.3.1.8 How to Render Master-Detail Navigation 14-19
14.3.2 Using the Navigation REST APIs 14-21
14.4 Using Portal Navigation to Create a Sitemap 14-22

15 Creating and Managing Skins

15.1 Overview of Skins 15-1
15.2 Creating a Skin in JDeveloper 15-2
15.2.1 What You Should Know About Skin Creation 15-2
15.2.2 How to Create a CSS File 15-3
15.2.3 How to Define Skin Selectors 15-4
15.3 Editing portal-skin.css 15-4
15.3.1 What You Should Know About portal-skin.css 15-4
15.3.2 How to Edit portal-skin.css 15-5
15.4 Enabling Runtime Administration of Skins 15-6
15.4.1 What You Should Know About Enabling Runtime Administration of Skins 15-6
15.4.2 How to Enable Runtime Administration of a Skin 15-8
15.4.3 How to Enable Further Development of Deployed Skins 15-8
15.4.3.1 Importing a Skin 15-9
15.4.3.2 Exporting a Skin 15-10
15.4.4 Creating a Skin in JDeveloper and Making it Available at Runtime 15-10
15.4.5 How to Create and Manage Skins at Runtime 15-11
15.5 Applying Skins to WebCenter Portal: Framework Applications 15-11
15.5.1 Applying a Skin at Design Time 15-11
15.5.2 Applying a Skin at Runtime 15-12

15.6 How to Create and Manage Spaces Skins 15-12
15.7 Deploying a Skin as a Separate Shared Library 15-13
15.7.1 About Skin Sharing 15-13
15.7.2 How to Deploy a Skin as a Separate Shared Library 15-14
15.8 Conditionally Changing Skins for Users 15-15
15.9 Troubleshooting Problems with Skins 15-16
15.10 Files Relates to Skins 15-17
15.10.1 adf-config.xml 15-17
15.10.2 trinidad-config.xml 15-18
15.10.3 trinidad-skins.xml 15-18

16 Creating and Managing Resource Catalogs

16.1 Overview of Resource Catalog 16-1
16.2 Creating a Custom Resource Catalog 16-4
16.2.1 How to Create a Custom Resource Catalog 16-4
16.2.2 How to Define Connections to Resources 16-6
16.2.3 How to Manage Content in the Resource Catalog 16-6
16.2.3.1 Adding a Folder 16-8
16.2.3.2 Adding a Link 16-8
16.2.3.3 Adding a Resource Catalog 16-11
16.2.3.4 Adding a Component 16-11
16.2.3.5 Adding a Custom Folder 16-13
16.2.3.6 Adding a Custom Content Provider 16-13
16.2.3.7 Dragging and Dropping a Resource into the Catalog 16-14
16.2.3.8 Editing a Resource 16-15
16.2.3.9 Deleting a Resource 16-15
16.2.3.10 Rearranging Resources in the Catalog 16-15
16.2.4 How to Expose Data Controls Created at Design Time 16-16
16.2.5 How to Expose Data Controls Created at Runtime 16-18
16.2.6 How to Control Visibility of Portlets in the Resource Catalog 16-18
16.2.7 How to Configure the Default Resource Catalog for an Application or Page 16-20
16.2.8 What You May Need to Know When Defining a Resource Catalog 16-20
16.2.9 How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows 16-21
16.3 Managing a Resource Catalog and Enabling Runtime Administration 16-22
16.3.1 Managing Resource Manager Entries 16-22
16.3.2 Enabling Further Development of Deployed Resource Catalogs 16-22
16.3.2.1 Exporting a Resource Catalog 16-23
16.3.2.2 Importing a Resource Catalog 16-23
16.3.3 Working with Spaces Resource Catalogs 16-24
16.4 Filtering Items in the Resource Catalog 16-24
16.4.1 Resource-Level Filtering 16-24
16.4.2 Catalog-Level Filtering 16-24
16.5 Configuring Multiple Resource Catalogs 16-26
16.5.1 How to Configure Multiple Resource Catalogs 16-26
16.5.2 What Happens at Runtime 16-28
16.6 Troubleshooting Problems with the Resource Catalog 16-29

17 Adding and Using the Resource Manager

17.1 Introduction to Runtime Resource Management 17-1
17.2 Working with Portal Resources in JDeveloper 17-3
17.2.1 How to Add a Resource to the Resource Manager 17-3
17.2.2 What You May Need to Know About the Content Directory 17-5
17.2.3 How to Update Portal Resource Properties 17-5
17.2.4 How to Remove a Resource from the Resource Manager 17-6
17.3 Enabling Runtime Administration of Your Portal 17-6
17.3.1 How to Add the Resource Manager Task Flow 17-6
17.3.2 How to Use the Resource Manager at Runtime 17-7
17.3.2.1 Accessing the Resource Manager 17-8
17.3.2.2 Creating a Resource 17-8
17.3.2.3 Uploading and Downloading a Resource 17-10
17.3.2.4 Editing a Resource 17-10
17.3.2.5 Copying a Resource 17-10
17.3.2.6 Setting Security for a Resource 17-11
17.3.2.7 Deleting a Resource 17-11
17.3.2.8 Showing or Hiding a Resource 17-11
17.3.2.9 Editing the Source Code of a Resource 17-12
17.3.2.10 Setting Properties on a Resource 17-13
17.3.2.11 Previewing a Resource 17-13
17.4 Enabling Round-Trip Development of Resources 17-13
17.4.1 How to Download a Resource Using the Resource Manager 17-16
17.4.2 How to Import a Portal Resource into JDeveloper 17-16
17.4.3 How to Export a Portal Resource from JDeveloper 17-17
17.4.4 How to Upload a Resource Using the Resource Manager 17-19

Part III Customizing Your Application and Extending Customization Options

18 Introduction to Composer

18.1 Application and User Customizations 18-2
18.2 View and Edit Modes of a Page 18-2
18.3 Customizing Capabilities in Page View Mode 18-3
18.3.1 Rearrange Components 18-3
18.3.2 Change the Layout 18-4
18.3.3 Expand and Collapse Components 18-5
18.4 Editing Capabilities in Design View in Page Edit Mode 18-6
18.4.1 Add Content 18-8
18.4.1.1 Add a Tab Set or a Tab to a Box Component 18-12
18.4.1.2 Add Box Components Adjacent to Existing Ones 18-13
18.4.1.2.1 Add Box Above 18-13
18.4.1.2.2 Add Box Below 18-14
18.4.1.2.3 Add Box Left 18-15
18.4.1.2.4 Add Box Right 18-15
18.4.2 Rearrange Page Content 18-16
18.4.3 Edit Component Properties and Parameters 18-17

18.4.4 Reset and Override Component Properties in the Current Layer 18-20
18.4.5 Edit Resource Strings 18-21
18.4.6 Show or Hide Components 18-27
18.4.7 Delete Components 18-28
18.4.8 Change the Layout 18-28
18.4.9 Edit Page Properties 18-29
18.4.10 Wire Components to Page Parameters 18-30
18.4.11 Reset Page 18-30
18.4.12 Reset Application Customizations on Detecting Errors 18-32
18.4.13 Create Labels On Saving Application Customizations 18-32
18.4.14 Manage Application Customizations 18-33
18.5 Editing Capabilities in Source View in Page Edit Mode 18-37
18.5.1 Rearrange Page Content Using Cut and Paste Options 18-40
18.5.2 Edit Content Inside a Task Flow 18-41
18.5.2.1 Reset Task Flow 18-43
18.5.3 Edit Content Inside a Declarative Component 18-44
18.5.4 Show or Hide Components 18-45
18.6 Composer Components 18-46
18.6.1 Page Customizable 18-47
18.6.2 Change Mode Link and Change Mode Button 18-48
18.6.3 Panel Customizable 18-48
18.6.4 Show Detail Frame 18-49
18.6.5 Layout Customizable 18-49
18.7 Security and Composer 18-50
18.7.1 Page and Task Flow Security 18-51
18.7.1.1 Task Flow Security 18-53
18.7.2 MDS Customization Restrictions 18-54
18.7.3 Component Action-Level Security 18-54

19 Enabling Runtime Creation and Management of Pages

19.1 Introduction to Page Creation and Management 19-1
19.2 Creating Pages and Task Flows 19-2
19.2.1 How to Create Pages 19-3
19.2.1.1 How to Add the Page - Create New Task Flow 19-4
19.2.1.2 Setting Security for the Page Service 19-4
19.2.1.3 How to Create Pages at Runtime 19-6
19.2.1.4 Structure of Pages Created at Runtime 19-6
19.2.1.5 How to Access Pages Created at Runtime 19-8
19.3 Defining Values for the Page - Create New Task Flow Parameters 19-8
19.3.1 How to Access Page - Create New Task Flow Parameters 19-8
19.3.2 Setting Scope in a Page - Create New Task Flow 19-10
19.3.3 Setting an Outcome Parameter 19-11
19.3.4 Specifying Styles 19-12
19.3.4.1 Out-of-the-Box Styles 19-12
19.3.4.2 Custom Styles 19-14
19.3.5 Specify an ADF Template 19-15
19.3.6 Showing a Command Link 19-15

19.3.7 Displaying an Image 19-16
19.3.8 Customizing the Label 19-16
19.4 How to Create Task Flow View Pages 19-17
19.5 Managing Pages 19-18
19.5.1 Using the Page Service Data Control to Manage Pages 19-18
19.5.2 Using the Page Service APIs to Manage Pages and Task Flows 19-18
19.6 Introduction to Custom Styles and Templates 19-19
19.6.1 How to Create Templates for Pages Created at Runtime 19-19
19.6.1.1 How to Create a Page or Page Fragment Style 19-19
19.6.1.2 How to Edit Create Page Dialog Styles 19-24
19.6.2 Using ADF Templates 19-25
19.6.2.1 How to Specify the ADF Template Name Using the Task Flow Parameter 19-25
19.6.2.2 How to Specify the ADF Template Name Using API Parameters 19-26
19.6.3 Creating Styles for the Create Page Dialog 19-26
19.6.3.1 How to Create a Style for the Create Page Dialog 19-26
19.6.3.2 How to Reference a New Style from the Page - Create New Task Flow 19-27
19.7 Customizing Page Service Views 19-28
19.7.1 Rendering Pages with ADF Faces Components 19-29
19.7.1.1 Rendering Pages as Tabs Using ADF Faces Components 19-29
19.7.1.1.1 Rendering Pages as Tabs Using af:navigationPane 19-29
19.7.1.1.2 Rendering Pages as Tabs Using af:panelTabbed 19-29
19.7.1.2 Rendering Pages as Links Using ADF Faces Components 19-30
19.7.1.2.1 Rendering Pages as Links Using af:commandLink 19-30
19.7.1.2.2 Rendering Pages as Links Using af:goLink 19-30
19.7.1.3 Rendering Pages as Image Links Using ADF Faces Components 19-31
19.7.2 Managing User Security on Pages and Task Flows 19-32
19.8 Advanced Information for the Page Service 19-32
19.8.1 Using the Page Service Java APIs 19-32
19.8.1.1 Configuration Settings Required for Using Page Service APIs 19-33
19.8.1.2 Introduction to the Page Service APIs 19-33
19.8.1.3 How to Set up Your Application to Use the Page Service APIs 19-35
19.8.1.4 Example: How to Create a Page 19-37
19.8.2 Using the Page Service Data Control 19-38
19.8.2.1 How to Add the Page Service Data Control 19-39
19.8.2.2 How to View, Edit, and Delete Pages at Runtime 19-42
19.9 Page Service Samples 19-42

20 Enabling Runtime Editing of Pages Using Composer

20.1 Designing Editable Pages Using Composer Components 20-1
20.1.1 How to Create a Customizable Page 20-2
20.1.2 How to Enable Runtime Customization Using a Page Customizable 20-3
20.1.3 How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button 20-4
20.1.4 How to Define Editable Areas of a Page Using Panel Customizable Components 20-6
20.1.5 How to Enable Layout Customization for a Page Using a Layout Customizable 20-7
20.1.6 How to Enable Component Customization Using Show Detail Frame Components 20-9

20.1.7 How to Create a Page Template for Creating Customizable Pages 20-10
20.1.8 How to Enable Customization in a Populated Page 20-11
20.1.9 What Happens When You Add Composer Components 20-12
20.1.10 What Happens at Runtime 20-13
20.1.11 What You May Need to Know When Designing Editable Pages 20-13
20.2 Designing Editable Pages Using Composer Components: Example 20-14
20.3 Populating Pages with Content 20-17
20.3.1 What You May Need to Know When Adding Content to the Page 20-18
20.3.1.1 Considerations for Adding Task Flows 20-18
20.4 Troubleshooting Composer Problems 20-18

21 Extending Runtime Editing Capabilities Using Composer

21.1 Overview of Extensibility Options 21-1
21.1.1 Composer Add-Ons 21-2
21.1.2 Composer Custom Property Panels 21-2
21.1.3 Composer Events Handlers 21-2
21.1.4 Drop Handlers 21-3
21.1.5 Component Property Filters 21-3
21.1.6 Customization Manager 21-3
21.1.7 Composer Toolbar Customization 21-3
21.1.8 Configuration Files 21-4
21.2 Creating Composer Add-Ons 21-4
21.2.1 How to Create and Register Add-Ons 21-5
21.2.1.1 Creating an Add-On Task Flow 21-5
21.2.1.2 Registering Add-Ons with Composer 21-7
21.2.1.3 Registering Add-Ons in adf-config.xml 21-8
21.2.2 What Happens at Runtime 21-9
21.2.3 How to Exclude Composer Default Add-Ons 21-9
21.2.4 How to Display the Customization Manager Add-On 21-10
21.2.5 How to Selectively Display Add-Ons 21-10
21.3 Creating Custom Property Panels 21-11
21.3.1 How to Create and Register Custom Property Panels 21-11
21.3.1.1 Creating a Custom Property Panel 21-12
21.3.1.2 Registering a Custom Property Panel for a Component 21-12
21.3.1.3 Registering a Custom Property Panel for a Task Flow 21-13
21.3.2 What Happens at Runtime 21-15
21.3.3 How to Override Default Property Panels 21-15
21.3.4 How to Exclude Default Property Panels 21-16
21.3.5 How to Selectively Render Property Panels 21-16
21.3.6 How to Display Properties and Parameters in a Custom Property Panel 21-16
21.4 Extending the Expression Builder 21-20
21.4.1 How to Extend the Expression Builder 21-21
21.4.2 How to Protect Expression Language 21-22
21.4.3 What Happens at Runtime 21-22
21.5 Configuring Custom LOVs or Pickers for Task Flow Parameters 21-23
21.5.1 How to Configure an LOV 21-28
21.5.1.1 Configuring a Static LOV 21-28

21.5.1.2 Configuring a Dynamic LOV 21-29
21.5.1.3 Configuring a Picker 21-30
21.5.1.3.1 Creating the Picker Task Flow 21-30
21.5.1.3.2 Registering the Picker for a Task Flow Parameter 21-34
21.5.1.4 Configuring an LOV from a Global List 21-35
21.5.2 What Happens at Runtime 21-36
21.6 Configuring Event Handlers for Composer UI Events 21-38
21.6.1 How to Create and Register Handlers for Composer UI Events 21-38
21.6.1.1 UI Events that Support Event Handler Registration 21-39
21.6.1.2 Creating a Save Event Handler: Example 21-40
21.6.1.3 Registering an Event Handler with Composer 21-41
21.6.2 What Happens When You Create and Register Event Handlers 21-41
21.6.3 Performing Additional Composer Event Handler Configurations 21-41
21.7 Configuring Drop Handlers in the Resource Catalog 21-43
21.7.1 How to Create and Register Java Drop Handlers 21-43
21.7.1.1 Creating a Drop Handler 21-43
21.7.1.2 Registering a Drop Handler with Composer 21-47
21.7.1.3 Adding an XML Component to the Resource Catalog 21-47
21.7.2 What Happens at Runtime 21-48
21.8 Defining Property Filters 21-49
21.8.1 How to Define Property Filters 21-50
21.8.2 What Happens at Runtime 21-50
21.8.3 How to Remove Property Filters 21-51
21.9 Enabling Parameter Support on the Customization Manager Task Flow 21-51
21.9.1 How to Set the Customization Manager Task Flow Parameters 21-53
21.9.2 What Happens When You Enable Parameter Support 21-54
21.10 Customizing the Composer Toolbar 21-54
21.10.1 How to Rearrange or Hide Toolbar Elements 21-56
21.10.2 What Happens at Runtime 21-56
21.10.3 How to Add New Sections to the Toolbar 21-57
21.10.4 How to Override a Toolbar Section to Display Custom Content 21-58
21.11 Enabling Direct Select in Design View 21-58
21.11.1 How to Enable and Configure Direct Select 21-59
21.11.2 What Happens at Runtime 21-61
21.12 Troubleshooting Problems with Composer Extensibility Features 21-62

22 Performing Advanced Composer Configurations

22.1 Enabling Custom Actions on a Show Detail Frame Component by Using Facets 22-1
22.2 Enabling Custom Actions on a Show Detail Frame Component By Using Facets: Example . 22-2
22.2.1 How to Create an ADF Task Flow 22-2
22.2.2 How to Include an Additional Actions Facet 22-4
22.2.3 How to Create a Redirection Page 22-4
22.2.4 How to Create Navigation Rules Between Pages 22-5
22.2.5 What Happens at Runtime 22-6
22.3 Enabling Custom Actions on a Task Flow 22-7
22.3.1 How to Enable Custom Actions Directly on a Task Flow 22-7

22.3.2 How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow ... 22-9
22.3.2.1 Defining Custom Actions at the Instance Level ... 22-9
22.3.2.2 Defining Custom Actions at the Global Level ... 22-10
22.3.2.3 Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window ... 22-12
22.3.3 What Happens at Runtime ... 22-14
22.3.4 How to Enable Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example ... 22-14
22.4 Enabling Custom Actions that Display on Task Flows in the Component Navigator ... 22-16
22.4.1 How to Configure Custom Actions in the Component Navigator ... 22-17
22.4.1.1 Defining the Logic for the Custom Action ... 22-17
22.4.1.2 Creating a JSPX Page Containing the Custom Action ... 22-18
22.4.1.3 Calling the JSPX Page from the Application Page Containing the Task Flow ... 22-19
22.4.2 What Happens at Runtime ... 22-20
22.5 Configuring a Keyboard Shortcut to Access Composer ... 22-21
22.5.1 How to Enable Linkless Entry Into Edit Mode ... 22-21
22.5.2 How to Configure an Event Handler for the Shortcut Key ... 22-22
22.5.3 What Happens at Runtime ... 22-22
22.6 Creating Event-Enabled Task Flows ... 22-22
22.7 Configuring an Application Page to Display in Source View by Default ... 22-22
22.7.1 How to Configure an Application Page to Display in Source View by Default ... 22-23
22.7.2 What Happens at Runtime ... 22-23
22.8 Disabling the Source View Menu for an Application ... 22-23
22.8.1 How to Disable the Source View Menu ... 22-23
22.8.2 What Happens at Runtime ... 22-24
22.9 Disabling Task Flow Zoom Capability ... 22-24
22.10 Applying Styles to Components ... 22-25
22.11 Configuring the Persistence Change Manager ... 22-27
22.11.1 Overview of the Default Change Manager Configuration ... 22-27
22.11.2 How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications ... 22-29
22.11.2.1 Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File ... 22-29
22.11.2.2 Adding Relevant Entries to the adf-config.xml File ... 22-29
22.12 Configuring Runtime Resource String Editing ... 22-30
22.12.1 Overview of the Resource String Editor ... 22-31
22.12.2 How to Enable the Resource String Editor in Your Application ... 22-32
22.12.3 How to Configure the Override Bundle ... 22-32
22.12.4 What Happens at Runtime ... 22-34
22.13 Troubleshooting Problems with Advanced Composer Configurations ... 22-34

23 Performing Composer-Specific MDS Configurations

23.1 Introduction to MDS ... 23-1
23.2 Using Composer Sandbox ... 23-2
23.2.1 How to Enable Composer Sandbox Creation ... 23-3
23.2.1.1 Updating Your Application's adf-config.xml File ... 23-3
23.2.1.2 Updating Your Application's web.xml File ... 23-4
23.2.1.3 Selecting a Database Store for Your Application ... 23-5
23.2.1.4 Enabling a Full Page Refresh at Runtime ... 23-5

23.2.2 What Happens at Runtime 23-5
23.2.3 How to Disable Sandbox for an Application 23-7
23.2.4 How to Destroy Stale Sandboxes 23-7
23.2.5 How to Enable Application Sandbox Creation 23-8
23.2.6 What Happens When You Enable Application Sandbox Creation 23-8
23.3 Adding Customization Layers to View and Edit Modes: Example 23-8
23.3.1 How to Add Composer to a JSF Page 23-9
23.3.2 How to Create a Custom SiteCC Tip Layer 23-10
23.3.3 How to Create a Custom UserCC Tip Layer 23-11
23.3.4 How to Implement the ComposerSessionOptionsFactory Class 23-13
23.3.5 How to Register the Implementation with Composer 23-14
23.3.6 How to Configure WebCenterComposerFilter 23-14
23.3.7 How to Redirect the Servlet to Enable Switch Between MDS Customization Layers 23-15
23.3.7.1 How to Create the AppNavigationUtils Class 23-16
23.3.7.2 How to Create AppCloseHandler 23-16
23.3.7.3 How to Register the AppCloseHandler 23-17
23.3.7.4 How to Create AppModeBean 23-17
23.3.8 What Happens at Runtime 23-18
23.4 Troubleshooting MDS-Related Problems with Composer 23-18

24 Modifying Default Security Behavior of Composer Components

24.1 Applying Component-Level Restrictions by Defining Customization Policies 24-1
24.1.1 How to Define Type-Level Customization Policies 24-2
24.1.2 How to Define Instance-Level Customization Policies 24-3
24.2 Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example 24-4
24.2.1 How to Configure ADF Security 24-5
24.2.2 How to Define Roles and Grant Privileges in the jazn-data.xml File 24-5
24.2.3 How to Customize the SessionOptions Object to Include Customization Policy 24-8
24.2.4 How to Register the Implementation with Composer 24-10
24.2.5 How to Configure WebCenterComposerFilter 24-10
24.2.6 How to Apply an Instance-Level Customization Restriction 24-10
24.2.7 What Happens at Runtime 24-11
24.3 Applying Tag-Level Security Using the customizationAllowed Attribute 24-13
24.3.1 How to Enable Application Customization on an Image Component 24-14
24.3.2 How to Restrict Customization on an Image Component 24-14
24.4 Applying Attribute-Level Security 24-14
24.4.1 How to Define Change Persistence at the Component Level 24-15
24.4.2 What Happens at Runtime 24-15
24.5 Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions 24-16
24.5.1 How to Add an enableSecurity Section to adf-config.xml 24-16
24.5.2 Defining Security at the Actions Category Level 24-17
24.5.3 Defining Security at the Actions Level 24-19
24.6 Implementing Task Flow Security 24-21
24.6.1 Granting Permissions on Task Flows 24-21

24.6.2 Enabling Task Flow Permission Check 24-23
24.7 Overriding Composer's Default Security Policies 24-23
24.7.1 How to Create a Custom Security Policy 24-24
24.7.2 How to Register a Custom Policy with Composer 24-25
24.7.3 What Happens at Runtime 24-25
24.8 Troubleshooting Problems with Composer Components Security 24-25

25 Customizing Task Flows

25.1 Preparing for Task Flow Customization 25-2
25.1.1 WebCenter Portal: Spaces: Create a New Task Flow Customization Application ... 25-2
25.1.2 WebCenter Portal: Framework Application: Enable Customization 25-2
25.2 Configuring the JDeveloper Customization Developer Role 25-3
25.3 Customizing Oracle WebCenter Portal Task Flows 25-5
25.3.1 Examples: Customizing Task Flows for WebCenter Portal: Spaces 25-5
25.3.1.1 Example: Customizing the Worklist Task Flow 25-5
25.3.1.2 Example: Customizing the Discussion Forums Task Flow 25-8
25.3.2 Examples: Customizing Task Flows for Custom Framework Applications 25-10
25.3.2.1 Example: Customizing the Document Library - List View Task Flow 25-11
25.4 Applying Task Flow Customizations to Deployed WebCenter Portal Applications 25-11
25.4.1 Creating a Metadata Deployment Profile for Custom WebCenter Portal Applications .. 25-12
25.4.2 Deploying Task Flow Customizations Directly from JDeveloper 25-13
25.4.3 Deploying Task Flow Customizations Using WLST 25-16
25.5 Removing Customizations from Deployed WebCenter Portal Applications 25-17
25.6 Catalog of WebCenter Portal Task Flows 25-17

Part IV Integrating and Publishing Content

26 Introduction to Integrating and Publishing Content

27 Managing Content Repository Connections

27.1 Overview of Content Adapters 27-1
27.2 Configuring Content Repository Connections 27-2
27.2.1 How to Create a Content Repository Connection Based on the Oracle Content Server Adapter 27-2
27.2.2 How to Create a Content Repository Connection Based on the Oracle Portal Adapter .. 27-8
27.2.3 How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint 27-11
27.2.3.1 Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint 27-11
27.2.3.2 Mapping Microsoft SharePoint Content and Services 27-13
27.2.4 How to Create a Content Repository Connection Based on the File System Adapter 27-15
27.2.5 What Happens When You Create a Repository Connection 27-16
27.2.6 What You May Need to Know When Creating a Repository Connection 27-17

27.2.6.1 What You Should Know About Using Identity Propagation and External Application Authentication Methods ... 27-17
27.2.6.2 What You Should Know About Oracle Portal ... 27-18
27.2.6.3 What You Should Know About Oracle WebCenter Adapter for SharePoint ... 27-19
27.3 Editing Content Repository Connections ... 27-21
27.3.1 How to Edit a Common Repository Connection ... 27-21
27.3.2 How to Edit a WebCenter Portal: Framework Application-Specific Content Repository Connection ... 27-23
27.4 Using an Existing Repository Connection for a New WebCenter Portal: Framework Application ... 27-23

28 Working with Content Data Controls

28.1 Overview of Content Data Controls ... 28-1
28.1.1 The getItems Method ... 28-2
28.1.2 The search Method ... 28-2
28.1.3 The advancedSearch Method ... 28-3
28.1.4 The getURI Method ... 28-4
28.1.5 The getAttributes Method ... 28-4
28.2 Configuring Content Data Controls for JCR Adapters ... 28-4
28.2.1 How to Configure a Content Repository Data Control ... 28-5
28.2.2 What Happens When You Configure a Content Repository Data Control ... 28-6
28.3 Editing Content Repository Data Controls ... 28-8
28.4 Securing a Content Repository Data Control ... 28-9
28.5 Integrating Content Using Content Data Controls ... 28-9
28.5.1 How to Publish Content As Links ... 28-10
28.5.1.1 Publishing Content As a Textual Link ... 28-11
28.5.1.2 Creating a Clickable Image to Link to a Document ... 28-15
28.5.2 What Happens at Runtime ... 28-16
28.5.3 How to Publish Content in a Table ... 28-17
28.5.3.1 Displaying Files and Folders in Read-Only Format ... 28-17
28.5.3.2 Displaying the Name Attribute As a Go Link ... 28-21
28.5.3.3 Configuring a Table to Show Only Files ... 28-24
28.5.4 What Happens at Runtime ... 28-26
28.5.5 How to Publish Folder Content in a Tree ... 28-27
28.5.5.1 Displaying Files and Folders in Read-Only Format ... 28-27
28.5.5.2 Displaying File Names As Hyperlinks ... 28-30
28.5.6 What Happens at Runtime ... 28-32
28.5.7 How to Add Search Capabilities to Content Repositories ... 28-33
28.5.7.1 Adding Simple Search Capabilities ... 28-33
28.5.7.2 Adding Advanced Search Capabilities ... 28-34
28.5.8 What Happens at Runtime ... 28-37
28.5.9 What You May Need to Know When Using Search Capabilities ... 28-38

29 Creating Content Presenter Display Templates

29.1 Adding Content Presenter to an Application ... 29-1
29.2 What Is a Content Presenter Display Template? ... 29-2
29.3 Using the Out-of-the-Box Display Templates ... 29-2

29.4 Creating Content Presenter Display Templates 29-3
29.4.1 Overview 29-4
29.4.2 Defining Single-Item Display Templates 29-4
29.4.3 Defining Multiple-Item Display Templates 29-8
29.4.4 Using EL Expressions to Retrieve Content Item Information 29-10
29.4.4.1 Retrieving Basic Information About a Content Item 29-10
29.4.4.2 Working with Content Item Properties and Values 29-11
29.4.4.3 Working with Content Item Icons and URLs 29-14
29.4.5 Discovering Content Type Property Names 29-14
29.4.6 Referencing External Files in Display Templates 29-16
29.4.7 Referencing Site Studio Region Elements in a Custom View 29-16
29.4.8 Using the Content Presenter Display Template 29-17
29.5 Using Content Presenter Display Templates in a Framework Application 29-18
29.5.1 Export a Content Presenter Display Template as a Portal Resource 29-18
29.5.2 Upload the New Content Presenter Display Template 29-21
29.5.3 Test the New Content Presenter Display Template 29-22
29.6 What Happens at Runtime? 29-24
29.6.1 Runtime Overview 29-24
29.6.2 Identifying Display Templates for Selected Content Items 29-24
29.7 Performance Considerations for Content Presenter Display Templates 29-25

30 Integrating the Documents Service

30.1 Introduction to the Documents Service 30-1
30.2 Setting Up Connections 30-2
30.3 Setting Security for the Documents Service 30-2
30.4 Using the Documents Service with Other WebCenter Services 30-4
30.5 Setting Parameters to Upload Files to Content Repositories 30-4
30.6 Using Adapters with the Documents Service 30-5

31 Adding Content Task Flows and Document Components to a Portal Page

31.1 Understanding the Content Presenter Task Flow 31-1
31.2 Understanding the Documents Service Task Flows 31-2
31.3 Understanding Document Components 31-10
31.4 Adding a Selected Folder or File to a Page 31-12
31.5 Adding a Content Task Flow to a Page 31-16
31.6 Modifying Content Task Flow Parameters 31-18
31.7 Content Task Flow Parameters 31-20
31.7.1 Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates 31-20
31.7.2 Document Explorer Task Flow Parameters 31-25
31.7.3 Document List Viewer Task Flow Parameters 31-27
31.7.4 Document Manager Task Flow Parameters 31-29
31.7.5 Document Navigator Task Flow Parameters 31-31
31.7.6 Folder Viewer Task Flow Parameters 31-33
31.7.7 Recent Documents Task Flow Parameters 31-35
31.7.8 Document Viewer Task Flow Parameters 31-36
31.7.9 Document Mini Properties Task Flow Parameters 31-38

31.7.10 Document Properties Task Flow Parameters 31-38
31.7.11 Rich Text Editor Task Flow Parameters 31-38
31.7.12 Document Upload Task Flow Parameters 31-40
31.7.13 Document Version History Task Flow Parameters 31-40
31.7.14 AutoVue Task Flow Parameters 31-41

32 Integrating Wikis and Blogs

32.1 Introduction to Wikis and Blogs 32-1
32.2 Requirements for Wikis and Blogs 32-1
32.3 Integrating Wikis 32-2
32.3.1 What You Should Know About the Document Explorer Task Flow 32-2
32.3.2 Adding the Document Explorer Task Flow to Provide the Wiki Functionality 32-2
32.3.3 Modifying Document Explorer Task Flow Parameters 32-3
32.3.4 What Happens at Runtime 32-3
32.3.5 Displaying Wiki Page Links Within Content Presenter 32-5
32.4 Integrating Blogs 32-5
32.4.1 Understanding the Blog Task Flows 32-5
32.4.2 Adding a Blog to a Page 32-7
32.4.2.1 Establishing a Selected Folder as a Blog Folder 32-8
32.4.2.2 Adding a Custom Blog to a Page Using the Blog Task Flows 32-9
32.4.3 What Happens at Runtime 32-10
32.4.4 Modifying Blog Task Flow Parameters 32-11
32.4.5 Blog Task Flow Parameters 32-12
32.4.5.1 Blog Archives Task Flow Parameters 32-12
32.4.5.2 Blog Banner Task Flow Parameters 32-12
32.4.5.3 Blog Digest Task Flow Parameters 32-12
32.4.5.4 Blog Recent Posts Task Flow Parameters 32-13
32.4.5.5 Blogs Task Flow Parameters 32-13

Part V Enabling Communication and Collaboration

33 Integrating the Announcements Service

33.1 Introduction to Announcements 33-1
33.1.1 Understanding the Announcements Service 33-1
33.1.2 Requirements for Announcements 33-2
33.1.3 What Happens at Runtime 33-2
33.2 Basic Configuration for the Announcements Service 33-3
33.2.1 Setting Up a Connection for Announcements 33-3
33.2.1.1 Announcements Connections 33-3
33.2.1.2 How to Set Up Connections for Announcements 33-4
33.2.2 Adding the Announcements Service at Design Time 33-7
33.2.2.1 Announcements Service Task Flows 33-7
33.2.2.2 How to Add Announcements to Your Page 33-8
33.2.2.3 How to Modify Announcements Service Task Flow Parameters 33-8
33.2.3 Setting Security for Announcements 33-11
33.3 Advanced Information for the Announcements Service 33-12

33.3.1 How to Add the Announcements - Quick View Task Flow 33-12
33.3.2 Customizing Announcements Views 33-13
33.3.3 Obtaining Space RSS News Feed URL for the Announcements Service 33-14

34 Integrating the Discussions Service

34.1 Introduction to Discussions 34-1
34.1.1 Understanding the Discussions Service 34-1
34.1.2 Requirements for Discussions 34-2
34.1.3 What Happens at Runtime 34-2
34.2 Basic Configuration for the Discussions Service 34-3
34.2.1 Setting up Connections for Discussions 34-3
34.2.1.1 Discussions Connections 34-3
34.2.1.2 How to Set Up Connections for Discussions 34-3
34.2.2 Adding the Discussions Service at Design Time 34-6
34.2.2.1 Discussions Task Flows 34-6
34.2.2.2 How to Add Discussions to a Page 34-7
34.2.2.3 How to Modify the Discussions Service Task Flow Parameters 34-8
34.2.3 Setting Security for Discussions 34-11
34.3 Advanced Information for the Discussions Service 34-12
34.3.1 Adding the Discussions - Popular Topics Task Flow 34-12
34.3.2 Adding the Discussions - Recent Topics Task Flow 34-13
34.3.3 Adding the Discussions - Watched Forums Task Flow 34-13
34.3.4 Adding the Discussions - Watched Topics Task Flow 34-14
34.3.5 Adding the Discussions - Quick View Task Flow 34-14
34.3.6 Obtaining Space RSS News Feed URL for the Discussions Service 34-15
34.3.7 Using Custom Discussions APIs 34-15
34.3.8 Using the Discussions Service REST APIs 34-15
34.3.8.1 Discussions Entry Point 34-16
34.3.8.2 Discussions Resource Type Taxonomy 34-16
34.3.8.3 Security Considerations 34-16
34.3.8.4 Discussions Resource Types 34-16
34.3.8.4.1 urn:oracle:webcenter:discussions:forums 34-17
34.3.8.4.2 urn:oracle:webcenter:discussions:forum 34-17
34.3.8.4.3 urn:oracle:webcenter:discussions:forum:topics 34-19
34.3.8.4.4 urn:oracle:webcenter:discussions:forum:topic 34-20
34.3.8.4.5 urn:oracle:webcenter:discussions:forum:topic:messages 34-22
34.3.8.4.6 urn:oracle:webcenter:discussions:forum:topic:message 34-23
34.3.9 Troubleshooting the Discussions Service 34-24

35 Integrating the Instant Messaging and Presence Service

35.1 Introduction to the IMP Service 35-1
35.1.1 Understanding the IMP Service 35-1
35.1.2 Requirements for IMP 35-2
35.2 Basic Configuration for the IMP Service 35-2
35.2.1 Setting up Connections for the IMP Service 35-3
35.2.1.1 IMP Service Connections 35-3

35.2.1.1.1 Mapping User Names to IM Addresses 35-3
35.2.1.2 How to Set Up Microsoft LCS Connections for the IMP Service 35-4
35.2.1.3 How to Set Up Microsoft OCS and Microsoft Lync Connections for the IMP Service 35-7
35.2.2 Adding the IMP Service at Design Time 35-10
35.2.2.1 IMP Service Task Flows 35-10
35.2.2.2 How to Add the IMP Service to your Application 35-11
35.2.3 Setting Security for the IMP Service 35-12
35.3 Advanced Information for the IMP Service 35-15
35.3.1 Enabling Network Presence 35-15
35.3.1.1 Setting Up Yahoo! Messenger Presence 35-15
35.3.1.2 Setting Up Other Network Presence Providers 35-17
35.3.2 Customizing IMP Views 35-19
35.3.3 Troubleshooting the IMP Service 35-20

36 Integrating the Mail Service

36.1 Introduction to the Mail Service 36-1
36.1.1 Understanding the Mail Service 36-1
36.1.2 Requirements for the Mail Service 36-3
36.2 Basic Configuration for the Mail Service 36-3
36.2.1 Configuration Roadmap - Mail Service 36-4
36.2.2 Setting up Connections for the Mail Service 36-5
36.2.3 Adding the Mail Service at Design Time 36-11
36.2.3.1 Mail Service Task Flows 36-11
36.2.3.2 How to Add the Mail Service to your Application 36-11
36.2.3.3 How to Modify the Mail Service Task Flow Parameters 36-12
36.2.4 Setting Security for the Mail Service 36-13
36.3 Advanced Information for the Mail Service 36-14
36.3.1 Invoking the Mail Compose Page 36-14
36.3.2 Configuring the Number of Mails Displayed 36-15
36.3.3 Troubleshooting the Mail Service 36-16

37 Integrating the Polls Service

37.1 Introduction to Polls 37-1
37.1.1 Understanding the Polls Service 37-1
37.1.2 Requirements for Polls 37-2
37.1.3 What Happens at Runtime 37-2
37.2 Basic Configuration for the Polls Service 37-2
37.2.1 Setting up Connections for the Polls Service 37-3
37.2.2 Adding the Polls Service at Design Time 37-3
37.2.2.1 Polls Task Flows 37-3
37.2.2.2 How to Add the Quick Poll Task Flow to Your Page 37-3
37.2.2.3 How to Modify the Polls Service Task Flow Parameters 37-3
37.2.3 Setting Security for Polls 37-5
37.3 Advanced Information for the Polls Service 37-5
37.3.1 How to Add the Polls Manager Task Flow 37-5
37.3.2 How to Add the Take Polls Task Flow 37-6

37.3.3 How to Add the View Poll Results Task Flow 37-6
37.3.4 Using the Polls Service Data Controls 37-6
37.3.4.1 Take Poll Data Control 37-6
37.3.4.1.1 createResponse 37-6
37.3.4.1.2 createResponseForLatestAvailablePoll 37-6
37.3.4.1.3 getPublishedPolls 37-7
37.3.4.1.4 getResults 37-7
37.3.4.1.5 saveResponse 37-7
37.3.4.2 Polls Manager Data Control 37-7
37.3.4.2.1 addQuestion 37-8
37.3.4.2.2 addQuestion 37-8
37.3.4.2.3 closePoll 37-8
37.3.4.2.4 closePollAfterDate 37-8
37.3.4.2.5 closePollAfterDays 37-9
37.3.4.2.6 closePollAfterResponses 37-9
37.3.4.2.7 createPoll 37-9
37.3.4.2.8 deletePoll 37-9
37.3.4.2.9 editPoll 37-9
37.3.4.2.10 editPoll 37-10
37.3.4.2.11 findPoll 37-10
37.3.4.2.12 getPolls 37-10
37.3.4.2.13 getPollsByUser 37-10
37.3.4.2.14 getPollsByUserCount 37-10
37.3.4.2.15 publishPoll 37-10
37.3.4.2.16 publishPollOnDate 37-11

Part VI Working with the People Connections Service

38 Introducing the People Connections Service

38.1 Overview of the People Connections Service 38-1
38.1.1 People Connections Service Task Flows 38-2
38.1.1.1 Activity Stream 38-2
38.1.1.2 Connections 38-5
38.1.1.3 Feedback 38-6
38.1.1.4 Message Board 38-7
38.1.1.5 Profile 38-9
38.1.1.6 The Publisher Task Flow 38-11
38.1.2 People Connections Service Requirements 38-12
38.2 Troubleshooting the People Connections Service 38-12

39 Basic Configuration for the People Connections Service

39.1 How to Set Up a Database Connection for the People Connections Service 39-1
39.2 How to Add People Connections Task Flows to a Page 39-1
39.3 Setting Security for the People Connections Service 39-2
39.4 Establishing Site-Level Settings for People Connections Features 39-3
39.4.1 Understanding the People Connections Site-Level Setting Scripts 39-3

39.4.1.1 Preparing and Running settings-insert.sql 39-3
39.4.1.2 Preparing and Running settings-update.sql 39-4
39.4.2 Supported Site-Level Settings for People Connections Features 39-5
39.4.2.1 Activity Stream Site-Level Settings 39-5
39.4.2.2 Connections Site-Level Settings 39-6
39.4.2.3 Feedback Site-Level Settings 39-7
39.4.2.4 Message Board Site-Level Settings 39-8
39.4.2.5 Profile Site-Level Settings 39-9
39.4.3 Example: Configuring Connections to Accept Invitations Automatically 39-11

40 People Connections Task Flow Binding Parameters

40.1 How to Revise People Connections Task Flow Binding Parameters 40-1
40.2 People Connections Task Flow Binding Parameters 40-3
40.3 What You Should Know About the Activity Stream Advanced Query Option 40-12

41 Using People Connections Data Controls and Java APIs

41.1 Using People Connections Data Controls 41-1
41.1.1 Adding a Data Control to Your Project 41-1
41.1.2 Working with People Connections Management Data Control 41-3
41.1.2.1 connectionLists 41-4
41.1.2.2 groupedPeopleCounts 41-4
41.1.2.3 receivedInvitations 41-4
41.1.2.4 sentInvitations 41-5
41.1.2.5 acceptInvitation(String invitationId) 41-5
41.1.2.6 addConnectionsToList(List userids, String listName) 41-5
41.1.2.7 createConnectionList(String listName) 41-5
41.1.2.8 declineInvitation(String invitationId) 41-6
41.1.2.9 dropConnectionList(String listName) 41-6
41.1.2.10 getConnectionList(String connectionListName) 41-6
41.1.2.11 getConnections(String userid, String connectionListName, String filterPattern, String sortBy, int startIndex, int fetchSize) 41-7
41.1.2.12 getNumberOfConnections(String userid, String connectionListName, String filterPattern) 41-7
41.1.2.13 ignoreInvitation(String invitationId) 41-8
41.1.2.14 inviteUserForConnection(String userid, String invitationMessage, Collection connectionListNames) 41-8
41.1.2.15 isConnectionListModifiable(String listName) 41-8
41.1.2.16 isConnectionPartOfUnmodifiableList(String connecteeGuid) 41-8
41.1.2.17 removeConnection(String userid, boolean fromListOnly, String listName) 41-9
41.1.2.18 searchUsers(String filterPattern, int startIndex, int fetchSize) 41-9
41.1.2.19 updateConnectionListsMembership(String member, Collection addListIds, Collection removeListIds) 41-9
41.1.3 Working with Profile Data Control 41-10
41.1.3.1 Method: getProfile(String userId) 41-10
41.1.3.2 Method: getProfileForUpdate(String userId) 41-10
41.2 People Connections Service Java APIs 41-10

42 People Connections Service REST APIs

42.1 Activity Stream REST APIs 42-1
42.1.1 Activity Stream Entry Point 42-2
42.1.2 Activity Stream Resource Type Taxonomy 42-2
42.1.3 Activity Stream Security Considerations 42-2
42.1.4 Activity Stream Resource Types 42-2
42.2 Connections and Profile REST APIs 42-10
42.2.1 Connections and Profile Entry Point 42-10
42.2.2 Connections and Profile Resource Type Taxonomy 42-11
42.2.3 Connections and Profile Security Considerations 42-11
42.2.4 Connections and Profile Resource Types 42-11
42.3 Feedback REST APIs 42-23
42.3.1 Feedback Entry Point 42-23
42.3.2 Feedback Resource Type Taxonomy 42-24
42.3.3 Feedback Security Considerations 42-24
42.3.4 Feedback Resource Types 42-24
42.4 Message Board REST APIs 42-26
42.4.1 Message Board Entry Point 42-26
42.4.2 Message Board Resource Type Taxonomy 42-27
42.4.3 Message Board Security Considerations 42-27
42.4.4 Message Board Resource Types 42-27
42.5 Creating an Invitation 42-30
42.5.1 Creating an Invitation 42-30
42.5.2 Deleting an Invitation 42-32

Part VII Helping Users Find Content

43 Integrating the Links Service

43.1 Introduction to the Links Service 43-1
43.1.1 Understanding the Links Service 43-1
43.1.2 Requirements for the Links Service 43-3
43.1.3 What Happens at Runtime 43-3
43.2 Basic Configuration for the Links Service 43-4
43.2.1 Setting up Connections for the Links Service 43-4
43.2.2 Adding the Links Service at Design Time 43-4
43.2.2.1 Links Service Task Flows 43-4
43.2.2.2 How to Add the Links Service to your Application 43-4
43.2.3 Setting Security for the Links Service 43-6
43.2.4 Troubleshooting the Links Service 43-6
43.3 Advanced Information for the Links Service 43-6
43.3.1 Using the Links Service REST APIs 43-6
43.3.1.1 Links Entry Point 43-7
43.3.1.2 Links Resource Type Taxonomy 43-7
43.3.1.3 Security Considerations 43-7
43.3.1.4 Links Resource Types 43-7
43.3.1.4.1 urn:oracle:webcenter:links:results 43-7

44 Integrating the Tags Service

44.1 Introduction to the Tags Service 44-1
44.1.1 Understanding the Tags Service 44-2
44.1.2 Requirements for the Tags Service 44-2
44.1.3 What Happens at Runtime 44-3
44.2 Basic Configuration for the Tags Service 44-3
44.2.1 Setting up Connections for the Tags Service 44-3
44.2.2 Adding the Tags Service at Design Time 44-3
44.2.2.1 Tags Service Components 44-4
44.2.2.2 Tags Service Task Flows 44-4
44.2.2.3 How to Add the Tags Service to your Page 44-5
44.2.2.4 How to Modify the Tags Service Task Flow Parameters 44-7
44.2.3 Setting Security for the Tags Service 44-8
44.3 Advanced Information for the Tags Service 44-9
44.3.1 Using the Tags Service Java APIs 44-9
44.3.2 Using the Tags Service REST APIs 44-9
44.3.2.1 Tags Entry Point 44-10
44.3.2.2 Tags Resource Type Taxonomy 44-10
44.3.2.3 Security Considerations 44-10
44.3.2.4 Tags Resource Types 44-11
44.3.2.4.1 urn:oracle:webcenter:tagging:tags 44-11
44.3.2.4.2 urn:oracle:webcenter:tagging:tag 44-12
44.3.2.4.3 urn:oracle:webcenter:tagging:taggedItems 44-13
44.3.2.4.4 urn:oracle:webcenter:tagging:taggedItem 44-14
44.3.2.4.5 urn:oracle:webcenter:tagging:users 44-15
44.3.3 Optional Way to Show Tags on Pages 44-16
44.3.4 Using the Resource Action Handling Framework to Tag Custom Objects 44-20
44.3.5 Troubleshooting the Tags Service 44-21
44.3.5.1 Tag Center Results Do Not Appear When Tag Is Clicked 44-21
44.3.5.2 The Tag Center Does Not Display Tagged Documents 44-21
44.3.5.3 The Tagging Button Does Not Display On Page 44-21
44.3.5.4 Clicking a Tagged Resource Does Not Go To The Correct Page 44-22

45 Integrating the Search Service

45.1 Introduction to Search 45-1
45.1.1 Understanding the Search Service 45-1
45.1.1.1 Understanding Search with the Oracle SES Adapter 45-2
45.1.1.2 Understanding Search with WebCenter Portal's Search Adapters 45-2
45.1.1.3 Click Actions on Search Results 45-2
45.1.2 What Happens at Runtime 45-2
45.2 Basic Configuration for the Search Service 45-3
45.2.1 Configuration Roadmap - Oracle SES 45-3
45.2.2 Setting up Connections for Oracle SES 45-5
45.2.3 Adding the Search Service at Design Time 45-9
45.2.3.1 Search Service Task Flows 45-9
45.2.3.2 How to Add the Search Service to Your Page 45-9
45.2.3.3 How to Modify the Search Service Task Flow Parameters 45-10

45.2.4 Setting Security for the Search Service 45-13
45.3 Advanced Information for the Search Service 45-13
45.3.1 Adding the Search Task Flow 45-14
45.3.2 Adding the Search - Saved Searches Task Flow 45-14
45.3.3 Adding the Search Preferences Task Flow 45-14
45.3.4 Configuring Search with the Oracle SES Adapter 45-14
45.3.5 Configuring Search with WebCenter Portal's Search Adapters 45-16
45.3.6 Using the Search Service Java APIs 45-17
45.3.7 Using the Search Service REST APIs 45-17
45.3.7.1 Search Entry Point 45-18
45.3.7.2 Search Resource Type Taxonomy 45-18
45.3.7.3 Security Considerations 45-18
45.3.7.4 Search Resource Types 45-18
45.3.7.4.1 urn:oracle:webcenter:search:results 45-18
45.3.8 Using the Search Service Data Control 45-21
45.3.8.1 Search Data Control Methods, Attributes, and Classes 45-21
45.3.8.1.1 SearchServiceDCRows 45-21
45.3.8.1.2 SearchServiceDCRefiner 45-22
45.3.8.1.3 SearchServiceDCRefinement 45-22
45.3.8.2 Integrating the Search Data Control 45-22
45.3.9 Building Adapters for the Search Service 45-22
45.3.9.1 How to Add a Search Source 45-23
45.3.9.2 How to Register a Custom Adapter 45-28
45.3.9.3 Search Adapter Attributes 45-28
45.3.9.3.1 Searchable Attributes 45-28
45.3.9.3.2 Keywords 45-28
45.3.9.3.3 Date Condition 45-29
45.3.9.3.4 Complex Date Condition 45-29
45.3.9.3.5 Person Condition 45-29
45.3.9.3.6 Complex Person Condition 45-29
45.3.9.3.7 Access 45-30
45.3.9.3.8 Selectable Attributes 45-30
45.3.9.4 What Happens at Runtime 45-31
45.3.10 Customizing the Search UI Without WebCenter Portal's Search Service APIs 45-31
45.3.11 Troubleshooting the Search Service 45-33

46 Integrating the Activity Graph Service

46.1 Introduction to the Activity Graph Service 46-1
46.1.1 Understanding the Activity Graph Service 46-1
46.1.2 Requirements for the Activity Graph Service 46-7
46.1.3 What Happens at Runtime 46-8
46.2 Basic Configuration for the Activity Graph Service 46-8
46.2.1 Configuration Roadmap for the Activity Graph Service 46-8
46.2.2 Setting Up Connections for the Activity Graph Service 46-10
46.2.3 Adding the Activity Graph Service at Design Time 46-11
46.2.3.1 Activity Graph Service Task Flows 46-11
46.2.3.1.1 The Recommended Connections Task Flow 46-11

46.2.3.1.2 The Similar Spaces Task Flow 46-12
46.2.3.1.3 The Similar Items Task Flow 46-13
46.2.3.2 How to Add Activity Graph Service Task Flows to a Page 46-14
46.2.3.3 How to Modify Activity Graph Service Task Flow Parameters 46-15
46.2.3.4 Activity Graph Service Task Flows and Task Flow Parameters 46-16
46.3 Advanced Information for the Activity Graph Service 46-17
46.3.1 Using the Activity Graph Data Control 46-18
46.3.2 Using the Activity Graph Service REST APIs 46-19
46.3.2.1 Activity Graph Entry Point 46-19
46.3.2.2 Activity Graph Resource Type Taxonomy 46-19
46.3.2.3 Security Considerations 46-20
46.3.2.4 Activity Graph Resource Types 46-20
46.3.2.4.1 urn:oracle:webcenter:activitygraph:recommendations 46-20
46.3.2.4.2 urn:oracle:webcenter:activitygraph:recommendations:recommendation . 46-22
46.3.2.4.3 urn:oracle:webcenter:activitygraph:items 46-22
46.3.2.4.4 urn:oracle:webcenter:activitygraph:items:item 46-23
46.4 Extending the Activity Graph Service 46-24
46.4.1 Defining Custom Node Classes 46-25
46.4.2 Defining Custom Actions 46-26
46.4.3 Defining Custom Similarity Calculations 46-28
46.4.4 Defining Custom Rank Calculations 46-29
46.4.5 Registering Custom Activity Providers 46-31
46.4.6 Registering Custom QRPPs 46-32
46.5 Troubleshooting the Activity Graph Service 46-33
46.5.1 Troubleshooting the Activity Graph Service Task Flows (All Task Flows) 46-33
46.5.2 Troubleshooting the Recommended Connections Task Flow 46-34
46.5.3 Troubleshooting the Similar Spaces Task Flow 46-34
46.5.4 Troubleshooting the Similar Items Task Flow 46-34

Part VIII Helping Users Keep Track

47 Integrating the Analytics Service

47.1 Introduction to the Analytics Service 47-1
47.1.1 Understanding the Analytics Service 47-1
47.1.2 Requirements for the Analytics Service 47-2
47.1.3 What Happens at Runtime 47-2
47.2 Basic Configuration for the Analytics Service 47-3
47.2.1 Configuration Roadmap for the Analytics Service 47-3
47.2.2 Setting up Connections for the Analytics Service 47-5
47.2.2.1 How to Set Up a Connection to the Analytics Database 47-6
47.2.2.2 How to Set Analytics Collector Properties 47-6
47.2.2.3 How to Set Up a Connection to the Analytics Collector 47-7
47.2.3 Adding Analytics Event Code to Your Application 47-8
47.2.3.1 Including Event Code for Page Views 47-8
47.2.4 Configuring a Namespace for Analytics Customizations in MDS 47-9
47.2.5 Adding Analytics Task Flows at Design Time 47-9
47.2.5.1 Analytics Task Flows 47-10

47.2.5.2 How to Add Analytics Task Flows to a Page 47-10
47.2.5.3 How to Modify Analytics Task Flow Parameters 47-12
47.2.5.4 How to Allow End Users to Customize Analytics Task Flows at Runtime 47-13
47.2.5.5 Analytics Task Flows and Task Flow Parameters 47-14
47.2.6 Setting up Security for Analytics Task Flows and Usage Data 47-15
47.3 Building Analytics Reports 47-16
47.3.1 Using SQL Data Controls 47-16
47.3.2 SQL Statements for Out-of-the-Box Analytics Reports 47-18
47.3.2.1 Analytics SQL: WebCenter Portal Traffic 47-19
47.3.2.2 Analytics SQL: Page Traffic 47-20
47.3.2.3 Analytics SQL: Login Metrics 47-20
47.3.2.4 Analytics SQL: Space Traffic 47-20
47.3.2.5 Analytics SQL: Space Response Time 47-21
47.3.2.6 Analytics SQL: Portlet Traffic 47-21
47.3.2.7 Analytics SQL: Portlet Instance Traffic 47-21
47.3.2.8 Analytics SQL: Portlet Response Time 47-22
47.3.2.9 Analytics SQL: Portlet Instance Response Time 47-22
47.3.2.10 Analytics SQL: Search Metrics 47-22
47.3.2.11 Analytics SQL: Document Metrics 47-23
47.3.2.12 Analytics SQL: Wiki Metrics 47-23
47.3.2.13 Analytics SQL: Blog Metrics 47-23
47.3.2.14 Analytics SQL: Discussion Metrics 47-23
47.3.3 Sample Queries for User Metrics 47-24
47.3.3.1 Sample SQL: Filter by User Property 47-24
47.3.3.2 Sample SQL: Group by User Property 47-24
47.3.4 Sample Analytics Database Queries for Specific Metrics 47-24
47.3.4.1 Sample SQL: User Activities 47-25
47.3.4.2 Sample SQL: Space Activities 47-25
47.3.4.3 Sample SQL: Space Activities for a Specific Time Period 47-26
47.3.4.4 Sample SQL: Activity for a Service During a Specific Time Period 47-27
47.3.4.5 Sample SQL: Search Phrases 47-28
47.3.4.6 Sample SQL: Page Views 47-28

48 Integrating the Lists Service

48.1 Introduction to the Lists Service 48-1
48.1.1 Understanding the Lists Service 48-1
48.1.2 What Happens to the Lists Service at Runtime 48-1
48.2 Basic Configuration for the Lists Service 48-2
48.2.1 Setting Up Connections for the Lists Service 48-2
48.2.2 Adding the Lists Service at Design Time 48-2
48.2.2.1 Lists Service Task Flows 48-2
48.2.2.2 How to Add the Lists Service to your Application 48-3
48.2.3 Enabling Users to Work with Lists at Runtime 48-3
48.3 Adding Lists to a Resource Catalog 48-4
48.4 Using the Lists Service REST APIs 48-4
48.4.1 Entry Point for Lists 48-5
48.4.2 Lists Resource Type Taxonomy 48-5

48.4.3 Lists Security Considerations 48-5
48.4.4 Lists Resource Types 48-5
48.4.4.1 urn:oracle:webcenter:space:lists 48-6
48.4.4.1.1 Navigation Paths to lists 48-6
48.4.4.1.2 Supported Methods for lists 48-6
48.4.4.1.3 Resource Types Linked to from lists 48-11
48.4.4.2 urn:oracle:webcenter:space:list 48-11
48.4.4.2.1 Navigation Paths to list 48-12
48.4.4.2.2 Supported Methods for list 48-12
48.4.4.2.3 Resource Types Linked to from list 48-16
48.4.4.3 urn:oracle:webcenter:space:list:rows 48-16
48.4.4.3.1 Navigation Paths to rows 48-16
48.4.4.3.2 Supported Methods for rows 48-16
48.4.4.3.3 Resource Types Linked to from rows 48-22
48.4.4.4 urn:oracle:webcenter:space:list:row 48-22
48.4.4.4.1 Navigation Paths to row 48-22
48.4.4.4.2 Supported Methods for row 48-22
48.4.4.4.3 Resource Types Linked to from row 48-26
48.4.4.5 urn:oracle:webcenter:space:list:columns 48-26
48.4.4.5.1 Navigation Paths to columns 48-26
48.4.4.5.2 Supported Methods for columns 48-26
48.4.4.5.3 Resource Types Linked to from columns 48-30
48.4.4.6 urn:oracle:webcenter:space:list:column 48-30
48.4.4.6.1 Navigation Paths to column 48-30
48.4.4.6.2 Supported Methods for column 48-31
48.4.4.6.3 Resource Types Linked to from column 48-34
48.5 Troubleshooting the Lists Service 48-34

49 Integrating Notifications

49.1 Introduction to Notifications 49-1
49.1.1 Understanding Notifications 49-1
49.1.2 Requirements for Notifications 49-2
49.1.3 What Happens with Notifications at Runtime? 49-3
49.2 Basic Configuration for Notifications 49-4
49.2.1 Adding the Notifications Namespace 49-4
49.2.2 Creating a Notifications Connection to a Mail Server 49-5
49.2.3 Creating a Notifications Connection to a BPEL Server 49-6
49.2.4 Setting Application Defaults for Notifications 49-7
49.2.4.1 What You Should Know About Subscription Defaults 49-7
49.2.4.2 Setting Notifications Subscription Defaults 49-9
49.2.5 Adding Notifications Task Flows to Your Application 49-11
49.2.5.1 Notifications Task flows 49-11
49.2.5.2 How to Add a Notifications Task Flow to a Page 49-12
49.2.5.3 Subscription Preferences Task Flow Input Parameters 49-13
49.2.6 Setting Security for Notifications 49-13
49.3 Advanced Information for Notifications 49-14
49.3.1 Using Notifications Java APIs 49-14

49.3.1.1 Configuration Settings Required to Use Notifications Java APIs 49-14
49.3.1.2 Introduction to Notifications Java APIs 49-14
49.3.1.3 How to Set up Your Application to Use Notifications Java APIs 49-15
49.3.2 Using Notifications Data Controls 49-16
49.3.2.1 Adding a Data Control to Your Project 49-16
49.3.2.2 Understanding the Notification Subscription Data Control 49-17
49.3.2.3 Notification Subscription Data Control Methods and Attributes 49-17
49.3.2.3.1 Method: getUserSubscriptionPreferences 49-17
49.3.2.3.2 Method: getScopeSubscriptionPreferences 49-17
49.3.2.3.3 Method: saveUserSubscriptionPreferences 49-18
49.3.2.3.4 Method: saveScopeSubscriptionPreferences 49-18
49.3.2.3.5 Method: getSubscriptions 49-18
49.3.2.3.6 Method: unsubscribe 49-19
49.3.2.4 Notification Subscription Data Control Classes 49-19
49.3.2.4.1 SubscriptionPreference 49-19
49.3.2.4.2 Subscription 49-20

50 Integrating the Recent Activities Service

50.1 Introduction to the Recent Activities Service 50-1
50.1.1 Understanding the Recent Activities Service 50-2
50.1.2 What Happens at Runtime 50-2
50.2 Basic Configuration for the Recent Activities Service 50-3
50.2.1 Setting up Connections for the Recent Activities Service 50-3
50.2.2 Adding the Recent Activities Service at Design Time 50-3
50.2.3 Modifying Recent Activities Service Task Flow Parameters 50-7
50.2.4 Setting Security for the Recent Activities Service 50-7
50.3 Advanced Information for the Recent Activities Service 50-7
50.3.1 Refining the Behavior of the Recent Activities Service 50-7
50.3.2 Obtaining RSS News Feed URL for the Recent Activities Service 50-8

51 Integrating the Worklist Service

51.1 Introduction to the Worklist Service 51-1
51.1.1 Understanding Worklists 51-1
51.1.2 Requirements for the Worklist Service 51-2
51.1.3 What Happens at Runtime 51-2
51.2 Roadmap - Configuring the Worklist Service for WebCenter Portal: Framework Applications 51-3
51.3 Basic Configuration for the Worklist Service 51-5
51.3.1 Setting up Connections for the Worklist Service 51-5
51.3.1.1 Worklist Service Connections 51-6
51.3.1.2 How to Set Up Connections for the Worklist Service 51-6
51.3.2 Adding the Worklist Service at Design Time 51-8
51.3.2.1 Worklist Service Task Flows 51-8
51.3.2.2 How to Add the Worklist Service to your Application 51-8
51.3.3 Setting Security for the Worklist Service 51-8
51.4 Advanced Information for the Worklist Service 51-9

52 Integrating the Events Service

52.1 Introduction to the Events Service 52-1
52.1.1 Understanding the Events Service 52-1
52.1.2 Requirements for the Events Service 52-2
52.1.3 What Happens at Runtime 52-2
52.2 Basic Configuration for the Events Service 52-2
52.2.1 Configuration Roadmap for the Events Service 52-3
52.2.2 Setting Up a Connection for the Events Service 52-4
52.2.3 Adding the Events Service at Design Time 52-6
52.2.3.1 Events Task Flows 52-6
52.2.3.2 How to Add the Events Service to Your Application 52-7
52.2.4 Setting Security for the Events Service 52-8
52.3 Using the Events Service REST APIs 52-8
52.3.1 Events Entry Point 52-9
52.3.2 Events Resource Type Taxonomy 52-9
52.3.3 Security Considerations 52-9
52.3.4 Events Resource Types 52-9
52.3.4.1 urn:oracle:webcenter:events:gsEvents 52-9
52.3.4.2 urn:oracle:webcenter:events:gsEvent 52-10
52.3.4.3 urn:oracle:webcenter:events:gsCategories 52-12
52.3.4.4 urn:oracle:webcenter:events:gsCategory 52-13
52.4 Troubleshooting the Events Service 52-14

53 Integrating the RSS Service

53.1 Introduction to the RSS Service 53-1
53.1.1 Understanding the RSS Service 53-1
53.1.2 Requirements for the RSS Service 53-2
53.1.3 What Happens at Runtime 53-2
53.2 Basic Configuration for the RSS Service 53-3
53.2.1 Setting Up a Proxy Server for the RSS Service 53-3
53.2.2 Adding the RSS Service at Design Time 53-3
53.2.2.1 About RSS Viewer Task Flow 53-4
53.2.2.2 How to Add the RSS Viewer Task Flow to Your Application 53-4
53.2.2.3 How to Modify RSS Viewer Task Flow Parameters 53-5
53.2.3 Setting Security for the RSS Viewer Service 53-6

Part IX Extending Your WebCenter Portal Application

54 Using Oracle WebCenter Portal REST APIs

54.1 Introduction to REST 54-1
54.2 Understanding the Username-Based Security Token Encryption 54-2
54.3 Benefits of Using REST 54-2
54.4 Introduction to WebCenter Portal's REST APIs 54-3
54.5 Understanding the Link Model 54-4
54.5.1 The Resource Index 54-5
54.5.2 The Anatomy of a Link 54-7

54.5.2.1 Resource Type 54-8
54.5.2.2 Relationship 54-8
54.5.2.3 Capabilities 54-8
54.5.2.4 Media Type 54-9
54.5.2.5 Templates 54-9
54.6 Understanding Items Hypermedia 54-11
54.7 Navigating Hypermedia Using HTTP 54-11
54.8 Security Considerations for WebCenter Portal REST APIs 54-13
54.9 Security Considerations for CMIS REST APIs 54-13
54.10 Understanding Common Types 54-14
54.10.1 Common Types 54-14
54.10.1.1 personReference 54-14
54.10.1.2 groupSpaceReference 54-14
54.10.2 Portable Contact Types 54-15
54.10.2.1 name Portable Contact Type 54-15
54.10.2.2 address Portable Contact Type 54-15
54.10.2.3 organization Portable Contact Type 54-16
54.10.2.4 value Portable Contact Type 54-16
54.11 Managing Caches 54-16
54.12 Configuring a Proxy Server 54-16
54.13 WebCenter Portal's REST API Examples 54-17
54.13.1 Navigating the Message Board Hypermedia 54-17
54.13.1.1 Accessing the Resource Index 54-17
54.13.1.2 Reading Messages 54-18
54.13.1.3 Creating a New Message 54-19
54.13.1.4 Updating a Message 54-21
54.13.1.5 Deleting a Message 54-21
54.13.2 Displaying Activity Stream Data 54-22
54.13.3 Using the WebCenter People Connections Service REST APIs with Ext 54-24
54.13.3.1 Creating the HTML Page 54-24
54.13.3.2 Creating the JavaScript File 54-24
54.13.3.3 The Result 54-27
54.13.4 Updating User Status 54-27

55 Integrating Other Oracle Applications

55.1 Integrating Oracle and Other Applications in WebCenter Portal 55-1
55.2 Integrating Siebel Applications 55-2
55.2.1 Integrating Siebel Applications as Web Services 55-2
55.2.1.1 Preparing the Siebel Application 55-3
55.2.1.1.1 Creating an Inbound Web Service 55-3
55.2.1.1.2 Creating Operations for the Inbound Web Service 55-4
55.2.1.2 Consuming a Siebel Web Service in a Framework Application 55-5
55.2.1.2.1 Creating a Data Control Based on a Siebel Web Service 55-5
55.2.1.2.2 Creating a Page Flow Diagram 55-6
55.2.1.2.3 Adding Pages and Navigation to the Page Flow Diagram 55-6
55.2.1.2.4 Creating the Query Page 55-7
55.2.1.2.5 Creating the Results Page 55-8

55.2.1.2.6 Testing the Application 55-8
55.2.2 Integrating Siebel Applications Using the Siebel Web Engine 55-9
55.3 Integrating E-Business Suite Applications 55-9
55.3.1 Introduction to Integrating EBS Applications 55-10
55.3.1.1 Understanding EBS Integration 55-10
55.3.1.2 Requirements for Integrating EBS Applications 55-10
55.3.2 Integrating EBS Applications in a Framework Application 55-10
55.3.2.1 Preparing the EBS Application for Remote Access 55-11
55.3.2.2 Registering the EBS WSRP Producer in the Framework Application 55-11
55.3.2.3 Creating a JSF Page to Consume the Remote Producer 55-12
55.3.2.4 Testing the Framework Application 55-12
55.3.3 Integrating EBS Applications in Spaces 55-12
55.3.3.1 Preparing the EBS Portlet for Remote Access 55-13
55.3.3.2 Registering the EBS WSRP Producer in Spaces 55-13
55.3.3.3 Adding the EBS Portlet to a Spaces Page 55-13
55.3.3.4 Testing the Spaces Portlet Connection 55-14
55.4 Integrating JD Edwards Applications 55-14
55.4.1 Integrating JD Edwards Applications in a Framework Application 55-14
55.4.1.1 Preparing the JD Edwards Application for Remote Access 55-14
55.4.1.2 Registering the JD Edwards WSRP Producer in the Framework Application . 55-15
55.4.1.3 Creating the JSF Page to Consume the Remote Producer 55-15
55.4.1.4 Testing the Framework Application 55-16
55.4.2 Integrating JD Edwards Applications in Spaces 55-16
55.4.2.1 Registering the Producer 55-16
55.4.2.2 Adding the JD Edwards Portlet to a Spaces Page 55-16
55.4.2.3 Testing the Spaces Portlet Connection 55-17
55.5 Integrating PeopleSoft Applications 55-17
55.5.1 Introduction to Integrating PeopleSoft Applications 55-17
55.5.1.1 Understanding PeopleSoft Integration 55-17
55.5.1.2 Requirements for Integrating PeopleSoft Applications 55-18
55.5.2 Integrating PeopleSoft Applications in a Framework Application 55-18
55.5.2.1 Preparing the PeopleSoft Application for Remote Access 55-18
55.5.2.2 Registering the PeopleSoft WSRP Producer in the Framework Application ... 55-19
55.5.2.3 Creating the JSF Page to Consume the Remote (WSRP) Producer 55-20
55.5.2.4 Setting up WS-Security for PeopleTools 55-21
55.5.2.4.1 Configuring WS-Security for WSS10 SAML Token with Message Integrity 55-21
55.5.2.4.2 Configuring WS-Security for WSS10 SAML Token with Message Protection ... 55-21
55.5.2.4.3 Configuring WS-Security for WSS10 Username Token with Password 55-22
55.5.2.5 Testing the Framework Application 55-23
55.5.3 Integrating PeopleSoft Applications in Spaces 55-23
55.5.3.1 Registering the Producer 55-23
55.5.3.2 Adding the PeopleSoft Portlet to a Spaces Page 55-24
55.5.3.3 Testing the Spaces Portlet Connection 55-24
55.6 Integrating Oracle Business Intelligence Presentation Services 55-24
55.6.1 Introduction to Integrating Oracle Business Intelligence Presentation Services 55-25

55.6.1.1 Understanding Oracle Business Intelligence Presentation Services Integration 55-25
55.6.1.2 Requirements for Integrating Oracle Business Intelligence Presentation Services ... 55-25
55.6.2 Configuring Credentials for Connecting to the Oracle BI Presentation Catalog 55-26
55.6.2.1 Creating the BIImpersonateUser 55-26
55.6.2.2 Granting Permissions to BIImpersonateUser 55-26
55.6.3 Integrating Oracle Business Intelligence Presentation Services in Framework Applications 55-27
55.6.3.1 Installing the Required Business Intelligence Extensions 55-27
55.6.3.2 Creating a Framework Application that Integrates with Oracle BI Presentation Services 55-28
55.6.3.2.1 Creating the Framework Application and JSF Page 55-28
55.6.3.2.2 Configuring the Application to Display the Oracle BI EE Presentation Catalog 55-29
55.6.3.2.3 Enabling the BI ADF Task Flow Permission 55-30
55.6.3.3 Testing the Framework Application 55-31
55.6.3.3.1 Adding Oracle BI Objects to the Resource Catalog 55-31
55.6.3.3.2 Adding Business Intelligence Content at Runtime 55-32
55.6.3.3.3 Modifying a Business Intelligence Object's Prompt Values 55-32
55.6.3.3.4 Modifying a Business Intelligence Task Flow's Initialization Parameters . 55-32
55.6.3.4 Post-Deployment Connection Configuration 55-33
55.6.3.5 Securing Oracle BI EE Objects Included in a Framework Application 55-33
55.6.4 Integrating Oracle Business Intelligence Objects in WebCenter Portal: Spaces 55-33
55.6.4.1 Adding or Modifying a Presentation Services Connection After Deployment 55-33
55.6.4.2 Adding Oracle BI Objects to the Spaces Resource Catalog 55-35
55.6.4.3 Adding Oracle BI Content at Runtime 55-35
55.6.4.4 Modifying a Business Intelligence Object's Prompt Values 55-36
55.6.4.5 Modifying a Business Intelligence Task Flow's Initialization Parameters 55-36

56 Extending the Spaces Application Using JDeveloper

56.1 Downloading a Workspace for Spaces Development 56-1
56.2 Including Additional Shared Libraries in Spaces 56-3
56.2.1 What You Should Know About Shared Library Deployment for Spaces 56-4
56.2.2 Deploying Your Own Custom Code and Task Flows in Shared Libraries 56-5
56.2.3 Rebuilding the Spaces Shared Library List 56-6
56.2.3.1 Editing the Shared Library List 56-6
56.2.3.2 Setting Build and Deployment Options (config.properties) 56-8
56.2.3.3 Building and Deploying the Shared Library List (extend.spaces.webapp.war) 56-9
56.2.3.4 Reverting to a Previous Shared Library List 56-10
56.2.3.5 Troubleshooting Shared Library Deployment 56-11
56.3 Developing Resources for the Spaces Application 56-12
56.3.1 Using the WebCenterSpacesResources Project for Round-Trip Development 56-12
56.3.2 Opening and Exploring the WebCenterSpacesResources Project 56-14
56.3.3 Enabling Direct Uploads to the Spaces Application 56-16
56.3.4 Editing Spaces Resources in JDeveloper 56-17
56.3.4.1 Downloading a Spaces Resource Directly into JDeveloper 56-19

56.3.4.2 Importing a Spaces Resource into JDeveloper 56-20
56.3.4.3 How to Build Spaces Resources 56-22
56.3.5 Uploading Resources Directly to the Spaces Application 56-23
56.3.6 Exporting Spaces Resources to an Archive 56-25
56.3.7 Troubleshooting Resource Round-Trip Development 56-26

57 Integrating with Oracle WebCenter Portal: Spaces

57.1 Introduction to WebCenter Portal: Spaces 57-1
57.2 Exposing Spaces in WebCenter Portal Applications Using APIs 57-4
57.2.1 Introduction to Spaces APIs 57-4
57.2.2 Case Study 1: Purchasing Application Uses a Space to Evaluate Suppliers 57-5
57.2.3 Case Study 2: Customer Support Center Application Uses a Space to Discuss Customer Escalations 57-7
57.2.4 How to Set Up Your WebCenter Portal Application to Use the Spaces APIs 57-8
57.2.4.1 Verifying That the Spaces Application Is Up and Running 57-8
57.2.4.2 Setting Up the WebCenter Portal Application to Use Spaces APIs 57-9
57.2.4.3 Securing the Connection Between the Application and Spaces 57-11
57.2.4.4 Setting Up the Spaces Client Context 57-11
57.2.5 How to Provide Spaces Functionality in WebCenter Portal Applications 57-12
57.2.5.1 Managing Spaces and Space Template 57-13
57.2.5.1.1 Creating a Space 57-14
57.2.5.1.2 Creating a Custom Attribute 57-14
57.2.5.1.3 Creating a Space Template 57-15
57.2.5.1.4 Deleting a Space 57-15
57.2.5.2 Managing Space Membership 57-15
57.2.5.2.1 Adding Members to a Space 57-16
57.2.5.2.2 Inviting Users to Join a Space 57-17
57.2.5.2.3 Removing Members from a Space 57-17
57.2.5.2.4 Retrieving Role Information 57-18
57.2.5.3 Retrieving Information for Spaces and Space Templates 57-18
57.2.5.3.1 Retrieving a List of Spaces 57-19
57.2.5.3.2 Retrieving a List of Public Spaces 57-19
57.2.5.3.3 Retrieving a List of Discoverable Spaces 57-19
57.2.5.3.4 Retrieving Space Metadata 57-20
57.2.5.3.5 Retrieving a List of Space Templates 57-22
57.2.5.3.6 Retrieving Space Template Metadata 57-22
57.2.5.3.7 Retrieving the Spaces URL 57-23
57.2.5.3.8 Retrieving a Space URL 57-23
57.2.5.3.9 Retrieving RSS Feed URLs for Space Services 57-24
57.2.6 How to Handle Exceptions Raised by Spaces APIs 57-25
57.2.6.1 Providing Localized Error Messages 57-26
57.2.6.2 Listing the Error Stack 57-26
57.2.7 Finding More Information on Spaces APIs 57-27
57.2.8 Troubleshooting Issues with Spaces APIs 57-27
57.3 Using the WebCenter Portal: Spaces REST APIs 57-28
57.3.1 WebCenter Portal: Spaces Entry Point 57-28
57.3.2 WebCenter Portal: Spaces Resource Type Taxonomy 57-29

57.3.3 WebCenter Portal: Spaces Security Considerations 57-29
57.3.4 WebCenter Portal: Spaces Resource Types 57-29
57.3.4.1 urn:oracle:webcenter:spaces 57-30
57.3.4.2 urn:oracle:webcenter:siteresources 57-31
57.3.4.3 urn:oracle:webcenter:spaces:resource:templates 57-35
57.3.4.4 urn:oracle:webcenter:space 57-35
57.3.4.5 urn:oracle:webcenter:space:resourceindex 57-38
57.3.4.6 urn:oracle:webcenter:space:roles 57-39
57.3.4.7 urn:oracle:webcenter:space:role 57-39
57.3.4.8 urn:oracle:webcenter:space:icon 57-40
57.3.4.9 urn:oracle:webcenter:space:attributes 57-41
57.3.4.10 urn:oracle:webcenter:space:attribute 57-42
57.3.4.11 urn:oracle:webcenter:space:members 57-42
57.3.4.12 urn:oracle:webcenter:space:member 57-44
57.3.4.13 urn:oracle:webcenter:space:lists 57-44
57.3.4.14 urn:oracle:webcenter:space:list 57-46
57.3.4.15 urn:oracle:webcenter:space:list:rows 57-47
57.3.4.16 urn:oracle:webcenter:space:list:row 57-48
57.3.4.17 urn:oracle:webcenter:space:list:columns 57-50
57.3.4.18 urn:oracle:webcenter:space:list:column 57-51
57.4 Exposing Enterprise Applications in Spaces 57-52
57.4.1 Exposing WebCenter Portal Applications in Spaces 57-53
57.4.2 Exposing Oracle Applications in Spaces 57-53
57.4.3 Exposing Non-Oracle Applications in Spaces 57-53

Part X Working with Portlets and Portals

58 Overview of Portlets

58.1 Introduction to Portlets 58-1
58.1.1 Portlet Anatomy 58-2
58.1.2 Portlet Resources 58-4
58.1.2.1 JSF Portlets 58-4
58.1.2.2 Prebuilt Portlets 58-5
58.1.2.3 Parameter Form and Parameter Display Portlets 58-6
58.1.2.4 Web Clipping 58-7
58.1.2.5 OmniPortlet 58-8
58.1.2.6 Programmatic Portlets 58-9
58.1.2.7 Deciding Which Tool to Use 58-10
58.1.3 Portlet Modes 58-11
58.1.3.1 View Mode 58-11
58.1.3.2 Edit Mode 58-11
58.1.3.3 Edit Defaults Mode 58-12
58.1.3.4 Help Mode 58-12
58.1.3.5 About Mode 58-12
58.2 Portlet Technologies Matrix 58-12
58.2.1 General Suitability 58-15

58.2.2 Expertise Required ... 58-17
58.2.3 Deployment Type ... 58-18
58.2.3.1 PDK-Java Producers ... 58-18
58.2.3.2 WSRP Producers ... 58-19
58.2.3.3 Producer Architecture ... 58-20
58.2.4 Caching Style ... 58-21
58.2.5 Development Tool ... 58-22
58.2.6 Portlet Creation Style ... 58-22
58.2.7 User Interface Flexibility ... 58-24
58.2.8 Ability to Capture Content from Web Sites ... 58-24
58.2.9 Ability to Render Content Inline ... 58-25
58.2.10 Charting Capability ... 58-26
58.2.11 Public Portlet Parameter Support ... 58-26
58.2.12 Private Portlet Parameter Support ... 58-27
58.2.13 Ability to Hide and Show Portlets Based on User Privileges ... 58-27
58.2.14 Multilingual Support ... 58-28
58.2.15 Pagination Support ... 58-28
58.2.16 Authenticating to External Applications ... 58-28

59 Creating Portlets with the Oracle JSF Portlet Bridge

59.1 Introduction to the Oracle JSF Portlet Bridge ... 59-1
59.2 Creating a Portlet from a JSF Application ... 59-2
59.2.1 How to Create a JSF Portlet Based on a Page ... 59-2
59.2.2 How to Create a JSF Portlet Based on a Task Flow ... 59-5
59.2.2.1 Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog ... 59-5
59.2.2.2 Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog ... 59-7
59.2.3 How to Test a JSF Portlet Using the Integrated WebLogic Server ... 59-8
59.2.4 What Happens at Runtime ... 59-9
59.3 What You May Need to Know When Creating a JSF Portlet ... 59-9
59.3.1 General Guidelines ... 59-10
59.3.2 Portlet Guidelines ... 59-11
59.3.3 Security Guidelines ... 59-12
59.3.4 JSF Guidelines ... 59-13
59.3.5 Oracle ADF Guidelines ... 59-13
59.4 Copying a Runtime-Created Skin to a JSF Portlet Producer Application ... 59-16

60 Creating Portlets with the Portlet Wizard

60.1 Introduction to Java Portlets ... 60-1
60.1.1 Introduction to Standards-Based Java Portlets ... 60-1
60.1.2 Introduction to PDK-Java Portlets ... 60-3
60.2 Creating Java Portlets ... 60-4
60.2.1 How to Create a JSR 286 Java Portlet ... 60-5
60.2.2 What Happens When You Create a JSR 286 Java Portlet Using the JDeveloper Wizard 60-13
60.2.3 After You Create a JSR 286 Java Portlet Using the JDeveloper Wizard ... 60-14
60.2.4 How to Create a PDK-Java Portlet ... 60-14

60.2.5 What Happens When You Create a PDK-Java Portlet 60-20
60.2.6 After You Create an Oracle PDK-Java Portlet Using the JDeveloper Wizard 60-21
60.3 Troubleshooting Java Portlets 60-21
60.3.1 Cannot Access the Create Portlet Wizard 60-21
60.3.2 Cannot Add the Portlet Functionality that I Want to the Portlet 60-22

61 Coding Portlets

61.1 General Guidelines for Coding Java Portlets 61-1
61.1.1 Guidelines for Portlet Modes 61-1
61.1.1.1 Guidelines for View Mode 61-2
61.1.1.1.1 HTML Guidelines for Rendering Portlets 61-2
61.1.1.1.2 Cascading Style Sheet Guidelines for Rendering Portlets 61-3
61.1.1.2 Guidelines for Edit Mode 61-4
61.1.1.2.1 Guidelines for Edit Mode Operations 61-4
61.1.1.2.2 Guidelines for Buttons in Edit Mode 61-4
61.1.1.2.3 Guidelines for Rendering Personalization Values 61-4
61.1.1.3 Guidelines for Edit Defaults Mode 61-4
61.1.1.3.1 Guideline for Edit Default Mode Options 61-5
61.1.1.3.2 Guidelines for Buttons in Edit Defaults Mode 61-5
61.1.1.3.3 Guidelines for Rendering Personalization Values 61-5
61.1.1.4 Guidelines for Help Mode 61-5
61.1.1.5 Guidelines for About Mode 61-5
61.1.1.6 Guidelines for Preview Mode 61-5
61.1.1.7 Guidelines for Full Screen Mode (PDK-Java Only) 61-6
61.1.2 Guidelines for Navigation within a Portlet 61-6
61.1.2.1 Intraportlet Links 61-7
61.1.2.2 Application Links 61-8
61.1.2.3 External Links 61-8
61.1.2.4 Internal/Resource Links 61-8
61.1.3 Guidelines for JavaScript 61-9
61.2 Enhancing JSR 286 Java Portlets 61-9
61.2.1 How to Edit the Portlet Deployment Descriptor File 61-9
61.2.2 How to Add Custom Portlet Modes 61-10
61.2.3 How to Access User Information 61-11
61.2.4 How to Customize the Runtime Environment 61-12
61.2.4.1 Supported Container Runtime Options 61-12
61.2.4.2 Setting Container Runtime Options for All Portlets in an Application 61-19
61.2.4.3 Setting Container Runtime Options for Individual Portlets 61-20
61.2.5 How to Use Portlet Events 61-21
61.2.5.1 Defining a Portlet Event at the Application Level 61-21
61.2.5.2 Adding a Processing Event to a Portlet 61-22
61.2.5.3 Adding a Publishing Event to a Portlet 61-23
61.2.6 How to Use Public Render Parameters 61-23
61.2.6.1 Defining a Public Render Parameter at the Application Level 61-24
61.2.6.2 Adding a Public Render Parameter to a Portlet 61-25
61.2.7 How to Add Personalization 61-26
61.2.7.1 Adding a Portlet Preference to a Portlet 61-26

61.2.7.2 Simple Portlet Personalization Example 61-27
61.2.8 How to Use Portlet Filters 61-29
61.2.8.1 Adding a Portlet Filter to an Application 61-29
61.2.8.2 Applying a Portlet Filter to a Portlet 61-30
61.2.9 How to Implement Portlet Caching 61-31
61.2.9.1 Implementing Expiry-Based Caching 61-32
61.2.9.2 Implementing Validation-Based Caching 61-33
61.2.10 How to Implement Export of Customizations (WSRP 2.0) 61-34
61.2.11 How to Implement Rewritten URLs for Resource Proxy 61-35
61.2.12 How to Implement Stateless Resource Proxying 61-35
61.2.13 How to Enable Java Object Cache for Preference Store Access 61-36
61.2.14 How to Implement Security for JSR 286 Portlets 61-36
61.3 Enhancing PDK-Java Portlets 61-36
61.3.1 Guidelines for PDK-Java Portlets 61-37
61.3.2 How to Add Portlet Modes 61-37
61.3.3 How to Implement Public Parameters 61-38
61.3.4 How to Implement Private Parameters 61-43
61.3.4.1 About Private Parameters 61-43
61.3.4.2 About Portlet URL Types 61-44
61.3.4.3 Building Links with the Portlet URL Types 61-45
61.3.4.4 Building Forms with the Portlet URL Types 61-46
61.3.4.5 Implementing Navigation within a Portlet 61-48
61.3.4.6 Restricting Navigation to Resources 61-51
61.3.5 How to Use JNDI Variables 61-51
61.3.5.1 Declaring JNDI Variables 61-52
61.3.5.1.1 Variable Types 61-52
61.3.5.1.2 Variable Naming Conventions 61-52
61.3.5.1.3 Examples 61-53
61.3.5.2 Setting JNDI Variable Values 61-53
61.3.5.3 Retrieving JNDI Variables 61-54
61.3.6 How to Access Session Information 61-55
61.3.7 How to Enhance Portlet Performance with Caching 61-58
61.3.7.1 Activating Caching 61-60
61.3.7.2 Adding Expiry-Based Caching 61-60
61.3.7.3 Adding Validation-Based Caching 61-61
61.4 Testing Portlet Personalization 61-61
61.5 Creating a Struts Portlet 61-61
61.5.1 How to Create a New Flow and View to Host the Portlet Actions 61-63
61.5.2 How to Create the New JSPs 61-64
61.5.3 How to Create a Portlet 61-64
61.5.4 How to Extend the Portlet to Add Business Logic 61-65
61.5.5 How to Register the Producer 61-65

62 Testing and Deploying Your Portlets

62.1 Introduction to Portlet Deployment Testing in a Development Environment 62-1
62.2 Testing a Portlet Producer Application on Integrated WebLogic Server 62-2
62.2.1 How to Test JSR 286 Portlets on Integrated WebLogic Server 62-3

62.2.2 What Happens When You Test JSR 286 Portlets on Integrated WebLogic Server ... 62-4
62.2.3 How to Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server . 62-5
62.2.4 What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server ... 62-7
62.3 Deploying a Portlet Producer Application to an Oracle WebLogic Managed Server Instance ... 62-7
62.3.1 How to Create Deployment Profiles ... 62-8
62.3.1.1 Creating a WAR Deployment Profile ... 62-8
62.3.1.2 Creating a Deployment Descriptor ... 62-10
62.3.2 How to Create and Provision a WebLogic Managed Server Instance ... 62-10
62.3.3 How to Create and Register the Metadata Service Repository ... 62-10
62.3.4 How to Create a WebLogic Managed Server Connection ... 62-10
62.3.5 How to Deploy a Portlet Producer Application to an Oracle WebLogic Managed Server Instance ... 62-11
62.3.6 What Happens When You Deploy a Portlet Producer Application to an Oracle WebLogic-Managed Portlet Server ... 62-12
62.4 Registering and Viewing Your Portlet ... 62-13

63 Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer

63.1 About Server Communication and the Proxy ... 63-2
63.1.1 The Pagelet Producer Proxy ... 63-2
63.1.1.1 About Pagelets and the Proxy ... 63-3
63.1.2 About HTTP and CSP ... 63-4
63.1.2.1 HTTP ... 63-4
63.1.2.2 CSP ... 63-5
63.2 Using Pagelets in Web Applications ... 63-5
63.2.1 Adding a Pagelet to a JSF Page in Oracle JDeveloper ... 63-6
63.2.1.1 Registering the Pagelet Producer with a WebCenter Portal: Framework Application ... 63-6
63.2.1.2 Adding a Pagelet to a JSF Page ... 63-7
63.2.1.3 Securing a Pagelet ... 63-8
63.2.2 Adding a Pagelet to a Web Page ... 63-8
63.2.2.1 Inserting Pagelets Using Javascript ... 63-8
63.2.2.1.1 Adding a Preference Editor Using Javascript ... 63-9
63.2.2.2 Accessing Pagelets Using REST ... 63-10
63.2.2.2.1 Pagelet Inject API ... 63-11
63.2.2.2.2 Data Retrieval APIs ... 63-13
63.2.2.3 Using Automatic Resizing with IFrames ... 63-14
63.2.3 Adding a Pagelet to a Page in WebCenter Portal: Spaces ... 63-15
63.3 Building Pagelets Using the Pagelet Producer ... 63-16
63.3.1 Using the Adaptive Pagelet Scripting Framework ... 63-16
63.3.1.1 Handling Structured HTTP Responses ... 63-17
63.3.1.2 Using Event Notification ... 63-19
63.3.1.2.1 Page-Level Events for Use with the Scripting Framework ... 63-23
63.3.1.3 Using In-Place Refresh ... 63-24
63.3.1.4 Using Session Preferences ... 63-25

63.3.1.5 Adaptive Pagelet Development Tips 63-27
63.3.2 Using Pagelet Producer Credential Mapping 63-28
63.3.3 Modifying Pagelet Functionality at Runtime 63-31
63.3.3.1 Creating Web Injectors 63-31
63.3.3.2 Creating Custom Parsers 63-33
63.3.4 Debugging Pagelets 63-34
63.3.4.1 HTTP Tracing 63-34
63.3.4.2 Transformation Tracing 63-36
63.4 Working with WSRP and Oracle JPDK Portlets 63-37
63.4.1 Using Pagelet Chrome 63-37
63.5 Working with OpenSocial Gadgets 63-38
63.5.1 Configuring Authentication 63-39
63.5.2 Storing User Preferences 63-39
63.5.3 Accessing WebCenter Portal Profile Information 63-39
63.5.4 Accessing a User's Activity Stream 63-40
63.5.5 Using Gadget Eventing 63-41
63.5.6 Example: Consuming an External OpenSocial Gadget 63-41
63.5.7 Example: Consuming a Local OpenSocial Gadget 63-41

64 Consuming Portlets

64.1 Introduction to Consuming Portlets 64-1
64.2 Registering Portlet Producers with a WebCenter Portal: Framework Application 64-2
64.2.1 How to Register a WSRP Portlet Producer 64-3
64.2.2 How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles 64-10
64.2.3 How to Register an Oracle PDK-Java Portlet Producer 64-11
64.2.4 How to Edit Portlet Producer Registration Settings 64-15
64.2.5 How to Test a Portlet Producer Connection 64-16
64.2.6 How to Refresh a Portlet Producer 64-16
64.2.7 How to Delete a Portlet Producer 64-16
64.3 Adding Portlets to a Page 64-17
64.3.1 How to Add a Portlet to a Page 64-17
64.3.2 What Happens When You Add a Portlet to a Page 64-18
64.3.3 What Happens at Runtime 64-20
64.4 Setting Attribute Values for the Portlet Tag 64-21
64.4.1 How to Set Attribute Values for the Portlet Tag Using the Property Inspector 64-22
64.4.2 How to Set Attribute Values for the Portlet Tag in Source Code 64-22
64.4.3 Common Attributes of the Portlet Tag 64-23
64.4.4 Appearance Attributes of the Portlet Tag 64-25
64.4.5 Behavior Attributes of the Portlet Tag 64-28
64.4.6 Portlet Modes Attributes of the Portlet Tag 64-29
64.4.7 Style Attributes of the Portlet Tag 64-30
64.4.8 Binding Attributes of the Portlet Tag 64-31
64.4.9 Customization Attributes of the Portlet Tag 64-31
64.4.10 Other Attributes of the Portlet Tag 64-31
64.4.11 What You May Need to Know About Maximize, Minimize, Restore, and Move 64-32
64.4.12 What You May Need to Know About IFRAMEs 64-33

64.5 Copying Portlets 64-34
64.5.1 How to Copy a Portlet and Place it on the Same Page 64-34
64.5.2 How to Copy a Portlet from One Application Page to Another 64-36
64.6 Deleting Portlets from Application Pages 64-38
64.7 Consuming WebCenter Services Portlets 64-39
64.7.1 Consuming WebCenter Services Portlets in Oracle Portal 64-40
64.7.2 Consuming WebCenter Services Portlets in Oracle WebLogic Portal 64-40
64.7.3 Consuming WebCenter Services Portlets in Oracle WebCenter Interaction 64-40
64.7.4 Setting WebCenter Services Portlets Parameters 64-41
64.7.4.1 Common WebCenter Services Portlets Parameters 64-42
64.7.4.2 Document Manager Portlet Parameters 64-42
64.7.4.3 Discussion Forums Portlet Parameters 64-43
64.7.4.4 Blogs Portlet Parameters 64-44
64.7.4.5 Lists Portlet Parameters 64-44
64.7.4.6 Polls Manager Portlet Parameters 64-44
64.7.4.7 Take Polls Portlet Parameters 64-45
64.7.4.8 Worklist Portlet Parameters 64-45
64.7.4.9 Announcements Portlet Parameters 64-45
64.7.4.10 Mail Portlet Parameters 64-45
64.7.4.11 Activity Stream Portlet Parameters 64-45
64.7.4.12 Tag Cloud Portlet Parameters 64-47
64.8 Adding the Portlet Producer Task Flow 64-47
64.9 Troubleshooting Portlets 64-48
64.9.1 Diagnostic Tools for Troubleshooting Portlets 64-48
64.9.1.1 Identify the Portlet Instance 64-49
64.9.1.2 Examine the Portlet Consumer Test Page 64-50
64.9.1.2.1 Access the Portlet Consumer Test Page 64-51
64.9.1.2.2 Locate the Portlet Producer 64-52
64.9.1.2.3 Locate and Run the Portlet Instance 64-52
64.9.1.2.4 Perform Sanity Checks 64-53
64.9.1.2.5 Check Consumer Configuration Entries 64-54
64.9.1.3 Examine the Producer Test Page 64-54
64.9.1.3.1 Access the Producer Test Page 64-56
64.9.1.3.2 Run the JSF Portlet as a Servlet 64-56
64.9.1.3.3 Examine the SOAP Monitor 64-56
64.9.2 Configuring the Portlet Logging File 64-57
64.9.3 Portlet Displays a Portlet Consumer Error 64-59
64.9.4 Portlet Displays a Portlet Timeout 64-59
64.9.5 Portlet Displays a Remote Portlet Communication Error 64-61
64.9.6 Portlet Displays a Remote Portlet Error 64-62

65 Creating Portlets with OmniPortlet

65.1 Introduction to OmniPortlet 65-1
65.2 Adding OmniPortlet to Your Application 65-2
65.3 Customizing OmniPortlet 65-3
65.4 Troubleshooting OmniPortlet Problems 65-4

66 Creating Content-Based Portlets with Web Clipping

66.1 Introduction to Web Clipping 66-1
66.2 Adding Web Clipping to Your Application 66-3
66.3 Integrating Authenticated Web Content Using Single Sign-On 66-4
66.4 Advanced Features of Web Clipping 66-6
66.4.1 Using the Web Clipping Transport API 66-7
66.4.1.1 Implementing the Web Clipping Transport API 66-7
66.4.1.2 Deploying the Web Clipping Transport API 66-7
66.4.2 Rewriting Image Links to Use a Resource Proxy 66-8
66.5 Current Limitations of Web Clipping 66-8
66.6 Troubleshooting Web Clipping 66-9

Part XI Delivering Personalized Content

67 Personalizing WebCenter Portal Applications

67.1 Introduction to Personalization 67-1
67.1.1 Personalization Architecture 67-2
67.1.2 Personalization Conductor 67-2
67.1.2.1 Personalized Scenarios 67-2
67.1.2.2 Personalized Application Flow 67-3
67.1.2.3 Property Set Integration 67-3
67.1.2.4 Provider Integration 67-3
67.1.3 Personalization Property Service 67-3
67.1.3.1 Property Service Architecture 67-3
67.1.3.2 Property Sets 67-4
67.1.3.3 Property Locators 67-4
67.1.4 Personalization Providers 67-4
67.1.4.1 Out-of-the-Box Providers 67-5
67.1.4.1.1 Property Service Provider 67-5
67.1.4.1.2 CMIS Provider 67-5
67.1.4.1.3 Activity Graph Provider 67-5
67.1.4.1.4 People Connections Locator 67-6
67.1.4.2 Custom Data Providers 67-6
67.1.4.3 Function Providers 67-6
67.2 Integrating Personalization in Your Application 67-6
67.2.1 Personalization Requirements 67-7
67.2.2 Authoring Personalized Scenarios in JDeveloper 67-7
67.2.2.1 Creating a Properties Namespace 67-8
67.2.2.2 Creating Property Sets and Property Definitions 67-9
67.2.2.3 Creating a New Scenario in the Scenario Editor 67-10
67.2.2.4 Specifying Scenario Flow Using Node Types 67-12
67.2.2.5 Defining Property Sets and Property Locators 67-13
67.2.2.6 Implementing and Calling Function Providers 67-13
67.2.2.7 Implementing and Calling Data Providers 67-14
67.2.2.8 Using the Out-of-the-Box Providers 67-16
67.2.2.8.1 Using the Property Service Provider 67-16

67.2.2.8.2 Using the CMIS Provider 67-16
67.2.2.8.3 Using the Activity Graph Provider 67-19
67.2.3 Creating File-Based Scenarios 67-23
67.2.4 Displaying Targeted Content at Runtime 67-27
67.2.4.1 Presenting Content with Content Presenter 67-27
67.2.4.2 Presenting Content with a Custom User Interface 67-27
67.3 Tutorial: Creating, Testing and Deploying a Simple Application 67-27
67.4 Extending Personalization 67-38
67.4.1 Adding Custom Providers 67-39
67.4.1.1 Extending the AnnotatedConfiguration Class 67-39
67.4.1.2 Implementing the Required Interfaces 67-39
67.4.1.3 Creating the Resource Files 67-47
67.4.1.4 Wiring the Data Provider to the Conductor 67-47
67.4.1.5 Invoking your Data Provider in a Scenario 67-47
67.4.2 Adding Custom Locators 67-48
67.4.2.1 Understanding Property Locators 67-48
67.4.2.2 Implementing the Required Interfaces 67-48
67.4.2.3 Implementing Security 67-49
67.4.2.4 Creating a Property Set Definition 67-49
67.4.2.5 Putting It All Together 67-49
67.4.2.6 Invoking the Property Locator 67-53

68 Using Personalization APIs Externally

68.1 Using Java APIs 68-1
68.2 Using the Property Service REST APIs 68-1
68.2.1 Accessing the Property Service's Resource Index 68-2
68.2.2 Creating a Namespace Using the Property Service 68-2
68.2.3 Viewing Property Set Definitions Within a Namespace Using the Property Service 68-3
68.2.4 Creating a Property Definition Within a Namespace Using the Property Service ... 68-3
68.2.5 Viewing Property Definitions Within a Namespace Using the Property Service 68-3
68.2.6 Creating a Property Set Definition Within a Namespace Using the Property Service 68-3
68.2.7 Creating a Property Set Within a Namespace Using the Property Service 68-3
68.2.8 Viewing a Property Set Within a Namespace Using the Property Service 68-3
68.2.9 Accessing the Conductor ResourceIndex 68-3
68.2.10 Accessing Namespaces Using the Conductor 68-4
68.2.11 Creating a Scenario Within a Namespace Using the Conductor 68-6
68.2.12 Viewing Scenarios Within a Namespace Using the Conductor 68-7
68.2.13 Executing a Scenario Using the Conductor 68-8
68.2.14 Viewing Data Providers Within a Namespace Using the Conductor 68-8
68.2.15 Viewing Detailed Data Provider Properties Within a Namespace Using the Conductor 68-10
68.2.16 Viewing Data Provider Connections Within a Namespace Using the Conductor . 68-12
68.2.17 Viewing Data Provider Connection Resources Within a Namespace Using the Conductor 68-14

68.2.18 Viewing Data Provider Connection Resources Method Within a Namespace Using the Conductor 68-19
68.2.19 Viewing Function Providers Within a Namespace Using the Conductor 68-20
68.3 Property Service REST APIs 68-24
68.3.1 Property Service Resource Index 68-24
68.3.1.1 ResourceIndex 68-24
68.3.2 Namespace CRUD 68-28
68.3.2.1 Create Namespace 68-28
68.3.2.2 Retrieve Namespace 68-30
68.3.2.3 Update Namespace 68-32
68.3.2.4 Delete Namespace 68-34
68.3.2.5 Retrieve all Namespaces 68-34
68.3.2.6 Delete all Namespaces 68-48
68.3.3 Property Definition CRUD 68-49
68.3.3.1 Create Property Definition - Create Integer Property Definition 68-49
68.3.3.2 Create Property Definition - Create Integer Array Property Definition 68-51
68.3.3.3 Retrieve Property Definition 68-54
68.3.3.4 Update Property Definition 68-56
68.3.3.5 Delete Property Definition 68-57
68.3.3.6 Retrieve all Property Definitions 68-58
68.3.3.7 Delete all Property Definitions 68-61
68.3.4 Property Set Definition CRUD 68-61
68.3.4.1 Create Property Set Definition 68-62
68.3.4.2 Retrieve Property Set Definition 68-66
68.3.4.3 Retrieve All Property Set Definitions 68-67
68.3.4.4 Delete all Property Set Definitions 68-68
68.3.5 Property Set CRUD 68-69
68.3.5.1 Create Property Set 68-69
68.3.5.2 Retrieve Property Set 68-75
68.3.5.3 Update Property Set 68-80
68.3.5.4 Delete Property Set 68-83
68.3.5.5 Retrieve all Property Sets 68-84
68.3.5.6 Delete all Property Sets 68-85
68.3.6 Property CRUD 68-86
68.3.6.1 Create Property 68-86
68.3.6.2 Retrieve Property 68-88
68.3.6.3 Update Property 68-90
68.3.6.4 Delete Property 68-91
68.3.6.5 Delete all Properties 68-92
68.3.7 JSON Payload 68-93
68.3.8 Exceptions 68-93
68.3.9 Query Parameters 68-93
68.3.10 Search Expressions 68-94
68.3.10.1 Format 68-94
68.3.10.2 Supported Operators 68-94
68.3.10.3 Supported Attribute Names 68-94
68.4 Using the Conductor REST APIs 68-96
68.4.1 Using the Data Provider Management APIs 68-96

68.4.1.1 Getting a Collection of Data Providers (GET) 68-97
68.4.1.2 Getting a Single Data Provider's Metadata (GET) 68-101
68.4.1.3 Getting Information About a Data Provider Connection (GET) 68-112
68.4.1.4 Getting Information about an Executable Resource for a Data Provider and Connection (GET) 68-115
68.4.1.5 Getting Method Information for an Executable Resource for a Data Provider and Connection (GET) 68-124
68.4.1.6 Getting Entire Data Provider Information with Details Projection (GET) 68-127
68.4.2 Using the Function Provider Management APIs 68-145
68.4.2.1 Getting a Collection of Function Providers (GET) 68-145
68.4.2.2 Getting Metadata for a Single Function Provider (GET) 68-171
68.4.3 Using the Namespace Management APIs 68-173
68.4.3.1 Getting a Collection of Namespaces (GET) 68-173
68.4.3.2 Retrieving a Single Namespace (GET) 68-177
68.4.3.3 Creating a Namespace (POST) 68-179
68.4.3.4 Deleting a Namespace (DELETE) 68-181
68.4.4 Using the Scenario Management APIs 68-182
68.4.4.1 Getting a Collection of Scenarios (GET) 68-182
68.4.4.2 Retrieving a Single Scenario (GET) 68-184
68.4.4.3 Creating a Scenario (POST) 68-187
68.4.4.4 Updating a Scenario (PUT) 68-190
68.4.4.5 Deleting a Scenario (DELETE) 68-193
68.4.4.6 Execute a Scenario (POST) 68-194
68.5 Calling Personalization Client Services Using ELs 68-195
68.5.1 Calling Personalization Client Services from JSF Pages 68-195
68.5.2 Calling Personalization Client Services from JSP Pages 68-196
68.5.3 Personalization Context Object Method Reference 68-196
68.5.4 Enabling Single Sign-on 68-198

Part XII Completing Your WebCenter Portal Application

69 Securing Your WebCenter Portal: Framework Application

69.1 Introduction to WebCenter Portal: Framework Application Security 69-2
69.2 Creating an Application Role 69-2
69.3 Configuring ADF Security 69-3
69.3.1 Configuring ADF Security Settings 69-3
69.3.2 Automated Security Grants for WebCenter Portal Services 69-8
69.4 Using the Role Manager Task Flow 69-16
69.5 Using the Enterprise Role Member Task Flows 69-17
69.6 Using the Page Hierarchy Security Editor 69-17
69.6.1 Introduction to Page Hierarchy Security 69-17
69.6.1.1 Page Hierarchy Model 69-18
69.6.1.2 Page-level and Page Hierarchy Security 69-19
69.6.1.3 Inherited and Delegated Security 69-19
69.6.2 Building a Page Hierarchy 69-20
69.6.2.1 Adding a Page Hierarchy to a Navigation 69-20
69.6.2.2 Adding Pages to a Page Hierarchy 69-20

69.6.2.3 Delegating Security 69-21
69.7 Creating Login Pages and a Login Component 69-22
69.8 Creating a Login Portlet 69-23
69.9 Adding Portlets to a Login Page 69-23
69.10 Creating a Self-Registration Page 69-24
69.10.1 Integrating the Spaces Self-Registration Task Flow 69-24
69.10.2 Building a Custom Self-Registration Page 69-25
69.10.3 Creating a Self-Registration Invitation Page 69-25
69.11 Creating a Reset Password Page 69-26
69.12 Configuring Basic Authentication for Testing Portlet Personalization 69-26
69.13 Working with External Applications 69-27
69.13.1 Using External Applications 69-28
69.13.1.1 Secured Service Connections 69-28
69.13.1.2 Automated Single Sign-On 69-30
69.13.2 Supplying User Credentials 69-31
69.13.3 Managing External Applications 69-31
69.13.3.1 Adding External Application Task Flows 69-32
69.13.3.2 Working with External Applications in Oracle JDeveloper 69-33
69.13.3.2.1 Registering an External Application in Oracle JDeveloper 69-33
69.13.3.2.2 Editing External Application Registration Details in Oracle JDeveloper .. 69-35
69.13.3.2.3 Deleting External Application Registration Details in Oracle JDeveloper 69-36
69.13.3.3 Working with External Applications in Enterprise Manager 69-36
69.13.3.4 Working with External Applications Using WLST 69-36
69.14 Registering Custom Certificates with the Keystore 69-36
69.15 Overriding Inherited Security on Portlets and Customizable Components 69-39
69.15.1 Portlets Security 69-41
69.15.1.1 Defining Security at the Actions Category Level 69-41
69.15.1.2 Defining Security at the Actions Level 69-42
69.16 Identity Propagation Mechanisms 69-44
69.17 Securing Identity Propagation Through WSRP Producers with WS-Security 69-45
69.17.1 Identity Propagation Without WS-Security 69-46
69.17.2 Identity Propagation with WS-Security 69-46
69.17.3 Configuring Security for WSRP Portlets 69-48
69.18 Implementing PDK-Java Portlet Security 69-49
69.18.1 Assumptions 69-49
69.18.2 Introduction to PDK-Java Portlet Security Features 69-49
69.18.2.1 Identity Propagation 69-49
69.18.2.2 Authorization 69-49
69.18.2.3 Message-level Security 69-49
69.18.3 Single Sign-On 69-50
69.18.3.1 External Application 69-50
69.18.3.2 No Application Authentication 69-51
69.18.4 Portlet Security Managers 69-51
69.18.4.1 Implementing Your Own Security Manager 69-52
69.18.5 Message Authentication 69-52
69.18.6 User Input Escape 69-53
69.18.6.1 Default Container Encoding 69-53

69.18.6.2 Escape Methods 69-53
69.19 Troubleshooting Security Issues 69-54
69.19.1 Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed 69-54

70 Deploying and Testing Your WebCenter Portal: Framework Application

70.1 Introduction to Oracle WebLogic Servers 70-1
70.2 Deploying a Framework Application to the Integrated WebLogic Server 70-2
70.2.1 Running a Framework Application in the Integrated WebLogic Server 70-3
70.2.2 Deploying a Framework Application to the Integrated WebLogic Server 70-4
70.3 Deploying a Framework Application to a WebLogic Managed Server 70-4
70.3.1 Deployment Roadmap 70-5
70.3.2 Packaging a Framework Application 70-6
70.3.2.1 Packaging Database Connections and Application Security 70-7
70.3.2.1.1 Packaging the Database Connections 70-7
70.3.2.1.2 Packaging Application Security Data 70-9
70.3.2.2 Creating Deployment Profiles 70-10
70.3.3 Preparing the Target Environment for Deployment 70-12
70.3.3.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance 70-12
70.3.3.2 Creating and Registering the Metadata Service Repository 70-12
70.3.3.3 Configuring the Target Environment 70-12
70.3.3.4 Creating a WebLogic Managed Server Connection 70-13
70.3.4 Deploying a Framework application to a Managed Server 70-13
70.3.4.1 Deploying to a Managed Server Using Local Data Sources 70-14
70.3.4.2 Deploying to a Managed Server Using Global Data Sources 70-20
70.4 Transporting Customizations Between Environments 70-24

71 Building Multilanguage Portals

71.1 Language Support in ADF Faces Components 71-2
71.2 Using Resource Bundles to Support Multiple Languages 71-2

Part XIII Appendixes

A Files for WebCenter Portal: Framework Applications

A.1 About Files A-1
A.2 Files Overview A-2
A.3 Files Related to WebCenter Portal: Framework Applications A-2
A.3.1 catalog-registry.xml A-2
A.3.2 default-catalog.xml A-3
A.3.3 default-navigation-model.xml A-3
A.3.4 index.html A-3
A.3.5 navigation-registry.xml A-3
A.3.6 pages.xml A-3
A.3.7 error.jspx A-3
A.3.8 home.jspx A-3
A.3.9 login.jspx A-3

A.3.10 navigation-renderer.jspx A-4
A.3.11 pageTemplate_globe.jspx A-4
A.3.12 pageTemplate_swooshy.jspx A-4
A.4 Files Related to JSR 286 Portlets A-4
A.4.1 portlet.xml A-4
A.4.2 oracle-portlet-tags.jar A-6
A.4.3 portlet_mode.jsp A-6
A.4.4 portlet_name.java A-7
A.4.5 portlet_nameBundle.jar A-7
A.4.6 web.xml A-7
A.5 Files Related to PDK-Java Portlets A-7
A.5.1 producer_name.properties A-7
A.5.2 _default.properties A-8
A.5.3 index.jsp A-8
A.5.4 portlet_name_modePage.jsp A-8
A.5.5 provider.xml A-8
A.5.5.1 provider.xml Syntax A-9
A.5.5.2 provider.xml Sample A-9
A.5.6 web.xml A-9
A.6 System Files A-10
A.6.1 adf-config.xml A-10
A.6.2 DataBindings.cpx A-13
A.6.3 faces-config.xml A-13
A.6.4 page_name.jspx A-13
A.6.5 PageDef.xml A-13
A.6.6 trinidad-config.xml A-15
A.6.7 web.xml A-15
A.6.8 weblogic-application.xml A-15
A.6.9 mds Subdirectory A-15
A.6.10 wsdl Subdirectory A-15
A.7 Files Related to Security A-16
A.7.1 connections.xml A-16
A.7.2 jazn-data.xml A-20
A.7.3 cwallet.sso A-22
A.7.4 jps-config.xml A-22
A.7.5 adf-config.xml A-22
A.8 Oracle JDeveloper Files A-23
A.8.1 pagetemplate-metadata.xml A-23
A.9 Files Related to Oracle WebCenter Portal Services A-23

B Composer Component Properties and Files

B.1 Composer Component Properties B-1
B.1.1 Page Customizable B-2
B.1.2 Change Mode Link and Change Mode Button B-5
B.1.3 Layout Customizable Component B-7
B.1.4 Panel Customizable Component B-11
B.1.5 Show Detail Frame Component B-17

B.1.6 Custom Action B-24
B.1.7 Show Property B-27
B.2 Composer-Specific Files and Configurations B-30
B.2.1 pe_ext.xml B-30
B.2.1.1 addon-config B-31
B.2.1.2 property-panels B-32
B.2.1.3 lov-config B-33
B.2.1.4 event-handlers B-34
B.2.1.5 drop-handlers B-34
B.2.1.6 filter-config B-35
B.2.1.7 elbuilder-config B-37
B.2.1.8 selection-config B-37
B.2.1.9 Sample pe_ext.xml File B-38
B.2.2 adf-config.xml B-39
B.2.2.1 page-editor-config B-40
B.2.2.2 addon-panels B-40
B.2.2.3 sandbox-namespaces B-41
B.2.2.4 session-options-factory B-41
B.2.2.5 enable-design-views B-41
B.2.2.6 allow-el and protect-el B-42
B.2.2.7 rcv-config B-42
B.2.2.8 customizableComponentsSecurity B-43
B.2.2.9 mds-config B-44
B.2.2.10 resource-string-editor B-45
B.2.2.11 enable-source-view B-45
B.2.2.12 enable-zoom B-45
B.2.2.13 persistent-change-manager B-45
B.2.2.14 taglib-config B-46
B.2.2.15 security-config B-46
B.2.2.16 Sample adf-config.xml File B-47
B.2.3 adf-settings.xml B-50
B.2.3.1 custComps-config B-50
B.2.4 web.xml B-50
B.2.4.1 CHANGE_PERSISTENCE Context Parameter B-50
B.2.4.2 WebCenterComposerFilter B-51
B.3 Composer Default Add-Ons and Property Panels B-51
B.3.1 Default Add-Ons B-51
B.3.2 Default Property Panels B-52
B.4 Composer Help Topic IDs B-54
B.5 Composer Components Style-Specific Properties B-56
B.5.1 Style Selectors for Composer Components B-56
B.5.1.1 Global Style Selectors B-56
B.5.1.2 Composer Status Indicator Style Selectors B-57
B.5.1.3 Page Customizable Style Selectors B-57
B.5.1.4 Layout Customizable Style Selectors B-59
B.5.1.5 Panel Customizable Style Selectors B-59
B.5.1.6 Show Detail Frame Style Selectors B-61

B.5.2 Style Attributes B-65
B.6 Customizable Components (HTML) Properties B-69
B.6.1 Panel Customizable (HTML) Component B-70
B.6.2 Show Detail Frame (HTML) Component B-72
B.6.3 Customizable Components (HTML) Style Selectors B-75
B.6.3.1 Panel Customizable (HTML) Style Selectors B-76
B.6.3.2 Show Detail Frame (HTML) Style Selectors B-76
B.6.3.3 Property Keys B-77
B.6.3.4 Icon Selectors B-78

C Resource Catalog Properties and Files

C.1 Configuration and Location of Catalog Definitions C-1
C.2 XML Schema C-2
C.3 Catalog Definition Attributes C-10
C.3.1 catalogDefinition C-10
C.3.2 folder C-11
C.3.3 url C-12
C.3.4 includeCatalog C-13
C.3.5 component C-14
C.3.6 customFolder C-15
C.3.7 customContent C-16
C.3.8 attributes C-17
C.3.9 attribute C-17
C.3.10 parameters C-19
C.3.11 schema C-19
C.3.12 resource C-20
C.4 Factory Classes Available for Adding Dynamic Resources to the Catalog C-21
C.4.1 XMLComponentFactory C-21
C.4.2 ContentUrlResourceFactory C-22
C.4.3 CustomFolderContextFactory C-22
C.4.4 DefaultFolderContextFactory C-22
C.4.5 ContentListFactory C-23
C.4.6 EnsembleContentProvider C-23

D Guidelines for Creating Task Flows to Be Used in Composer-Enabled Pages

D.1 Guidelines for Effective Geometry Management and Pagination D-1
D.1.1 Guideline 1: Create Task Flows that Flow D-2
D.1.1.1 Example 1: Oracle ADF Faces Table Component with Its autoHeightRows Attribute Set to a Specific Value D-2
D.1.1.2 Example 2: Oracle ADF Faces Iterator Component Inside a Panel Group Layout with a Scrolling Layout D-3
D.1.2 Guideline 2: Limit the Number of Records Displayed in a Task Flow D-3
D.1.3 Guideline 3: Specify a Minimum Height for the Task Flow D-5
D.1.4 Guideline 4: Include UI for Accessing Data Beyond the Display Limit D-5
D.2 Guidelines for Efficient Use of Task Flow Parameters and Customization and Personalization Options D-7
D.2.1 Guidelines for Implementing Task Flow Parameters D-10

D.2.2 Guidelines for Implementing Customization D-10
D.2.3 Guidelines for Implementing Personalization D-11
D.2.4 Guidelines for Implementing Implicit Personalization D-11
D.3 Example of a Framework Application Containing a Task Flow Created By Following the Guidelines D-12
D.3.1 The RecentPagesTaskFlow Project D-12
D.3.1.1 The recent-pages-task-flow-definition.xml File D-13
D.3.1.2 The mainView.jsff Fragment D-14
D.3.1.3 The simpleEditPopupView.jspx Page D-18
D.3.1.4 The advancedEditPopupView.jspx Page D-19
D.3.1.5 The recentPagesBean Java Bean D-21
D.3.1.6 The Preference Bean D-27
D.3.2 The WebPages Project D-30
D.3.2.1 The Welcome.jspx File D-31
D.3.2.2 The adf-config.xml File D-32
D.3.3 Runtime Behavior D-33
D.4 Conclusion D-34

E Additional Portlet Configuration

E.1 OmniPortlet Configuration Tips E-1
E.1.1 Configuring the OmniPortlet Producer to Access Data Outside a Firewall E-2
E.1.2 Configuring the OmniPortlet Producer to Access Other Relational Databases E-5
E.1.2.1 Installing DataDirect JDBC Drivers E-5
E.1.2.2 Registering DataDirect Drivers in OmniPortlet E-5
E.1.3 Configuring Portal Tools and Web Producers (Optional) E-8
E.2 Web Clipping Portlet Configuration Tips E-10
E.2.1 Web Clipping Repository Configuration E-10
E.2.1.1 Using Oracle Metadata Services (MDS) as the Web Clipping Repository E-12
E.2.1.2 Using an Oracle Database as the Web Clipping Repository E-13
E.2.1.2.1 Using the Database Schema Created by RCU E-13
E.2.1.2.2 Creating a Database Schema E-13
E.2.1.3 Configuring Web Clipping Repository in provider.xml E-14
E.2.1.4 Attributes and Child Tags of the repositoryInfo Tag E-15
E.2.2 HTTP or HTTPS Proxy Configuration E-17
E.2.3 Web Clipping Producer Security E-18
E.2.3.1 Adding Certificates for Trusted Sites E-18
E.2.3.2 Configuring Oracle Advanced Security for the Web Clipping Producer E-19
E.3 Hiding or Removing the WSRP Test Page E-20
E.4 Setting Up a Preference Store E-21
E.4.1 WSRP Producers E-21
E.4.2 PDK-Java Producers E-23
E.5 Portlet Preference Store Migration Utilities E-24
E.5.1 JPS Portlet Preference Store - PersistenceMigrationTool E-24
E.5.2 PDK-Java Portlet Preference Store - Migration and Upgrade Utilities E-27
E.5.2.1 Migration Mode E-28
E.5.2.2 Upgrade Mode E-29
E.5.3 Web Clipping Repository E-30

E.5.4 Moving a Producer E-31
E.6 Exporting and Importing Portlet Producers at Design Time E-31
E.6.1 Exporting Portlet Producers at Design Time E-31
E.6.2 Importing Portlet Producers at Design Time E-32
E.7 Migrating WebLogic Portal Portlets to WebCenter Portal E-33
E.7.1 Migrating Java Portlets from WebLogic Portal to WebCenter Portal E-33
E.7.2 Problems With Migrating WLP Portlets to a WebCenter Portal Application E-33
E.7.3 General Tips for Migrating WLP Portlets to a WebCenter Portal Portlet Producer Application E-34

F Reuse of Oracle Portal Components

F.1 Introduction to Oracle Portal Components F-1
F.2 Reusing Portlets F-2
F.2.1 How to Reuse JSR 286 and Oracle PDK-Java Portlets F-2
F.2.2 What You May Need to Know About Events F-3
F.2.3 What You May Need to Know About Mobile Portlets F-3
F.2.4 What You May Need to Know About the Portlet Chrome F-3
F.2.5 What You May Need to Know About Personalizations and Customizations F-3
F.2.6 What You May Need to Know About Oracle Portal System Resources F-3
F.2.7 What You May Need to Know About Partner and External Applications F-4
F.2.8 How to Use the Federated Portal Adapter to Reuse Database Portlets F-4
F.2.9 What You May Need to Know About Troubleshooting the Federated Portal Adapter .. F-6
F.2.9.1 On Registration F-6
F.2.9.2 During Runtime F-7
F.2.10 What You May Need to Know About Limitations of the Federated Portal Adapter . F-7
F.2.11 How to Reuse Oracle PDK-Java Producers from Earlier Oracle Application Server Versions F-8
F.2.11.1 Consuming a Portlet from Oracle Portal F-8
F.2.11.2 Redeploying PDK-Java Producers from Oracle Portal F-9
F.3 Reusing Items F-10

G Expression Language Expressions

G.1 ELs Related to Applications G-1
G.2 ELs Related to Security G-2
G.3 ELs Related to General Settings G-2
G.4 ELs Related to Application Resources G-3
G.5 ELs Related to Navigation G-6
G.6 ELs Related to WebCenter Portal Services G-12
G.7 ELs Related to the Documents Service G-14
G.8 ELs Related to the People Connections Service G-14
G.9 ELs Related to the Personalization Service G-17

H Manually Migrating a Framework Application

H.1 When is Migration Required? H-1
H.2 Anatomy of a Framework Application H-2
H.3 Overview of the Migration Procedure H-2

H.4 Preparing for the Migration H-2
H.5 Editing Configuration Files H-3
H.5.1 Updating DataBindings.cpx H-3
H.5.2 Updating weblogic.xml H-3
H.5.3 Updating faces-config.xml H-4
H.5.4 Updating trinidad-config.xml H-4
H.5.5 Updating web.xml H-5
H.5.6 Updating adf-config.xml H-7
H.6 Modifying the Project Configurations H-9
H.7 Creating Portal-Specific Files H-10
H.7.1 Create the Framework Application Folder Structure H-10
H.7.2 Create a Resource Registry File H-11
H.7.3 Create Default Resources H-11
H.7.3.1 Default Navigation Model File H-11
H.7.3.2 Navigation Registry File H-12
H.7.3.3 Resource Catalog File H-12
H.7.3.4 Resource Catalog Registry File H-13
H.7.3.5 Page Template File H-14
H.7.3.6 Navigation Renderer File H-14
H.7.3.7 Page Hierarchy File H-15
H.8 Applying Security Configurations H-16
H.8.1 Configure ADF Security H-16
H.8.2 Configure the jazn-data.xml File H-16

I WebCenter Portal Analytics Database Schema

I.1 Analytics Database Schema Overview I-1
I.1.1 Analytics Database Model Overview I-1
I.1.2 Analytics Database Table Overview I-3
I.1.2.1 Sample SQL Query: Finding Table Names Associated with Specific Events I-7
I.2 Analytics Fact Table Descriptions I-8
I.2.1 ASFACT_WC_PAGECRE_0 ..... I-8
I.2.2 ASFACT_WC_PAGEDEL_0 ..... I-8
I.2.3 ASFACT_WC_PAGEEDI_0 ..... I-9
I.2.4 ASFACT_WC_PAGETAG_0 ..... I-9
I.2.5 ASFACT_WC_PAGEVIE_0 ..... I-9
I.2.6 ASFACT_WC_DISCUSS_0 ..... I-10
I.2.7 ASFACT_WC_DISCUSS_1 ..... I-11
I.2.8 ASFACT_WC_DISCUSS_2 ..... I-11
I.2.9 ASFACT_WC_DISCUSS_3 ..... I-12
I.2.10 ASFACT_WC_DISCUSS_4 ..... I-12
I.2.11 ASFACT_WC_DISCUSS_5 ..... I-13
I.2.12 ASFACT_WC_DISCUSS_6 ..... I-13
I.2.13 ASFACT_WC_DISCUSS_7 ..... I-14
I.2.14 ASFACT_WC_DISCUSS_8 ..... I-14
I.2.15 ASFACT_WC_DISCUSS_9 ..... I-14
I.2.16 ASFACT_WC_DISCUSS_A I-15
I.2.17 ASFACT_WC_DISCUSS_B I-15

I.2.18 ASFACT_WC_DISCUSS_C I-16
I.2.19 ASFACT_WC_DOCLIB__0 ..... I-16
I.2.20 ASFACT_WC_DOCLIB__1 ..... I-17
I.2.21 ASFACT_WC_DOCLIB__2 ..... I-18
I.2.22 ASFACT_WC_DOCLIB__3 ..... I-18
I.2.23 ASFACT_WC_DOCLIB__4 ..... I-18
I.2.24 ASFACT_WC_DOCLIB__5 ..... I-19
I.2.25 ASFACT_WC_DOCLIB__6 ..... I-19
I.2.26 ASFACT_WC_DOCLIB__7 ..... I-20
I.2.27 ASFACT_WC_EVENT_C_0 ..... I-20
I.2.28 ASFACT_WC_EVENT_D_0 ..... I-21
I.2.29 ASFACT_WC_EVENT_E_0 ..... I-21
I.2.30 ASFACT_WC_GROUPSP_0 ..... I-22
I.2.31 ASFACT_WC_GROUPSP_1 ..... I-22
I.2.32 ASFACT_WC_GROUPSP_2 ..... I-23
I.2.33 ASFACT_WC_GROUPSP_3 ..... I-23
I.2.34 ASFACT_WC_LIST_CR_0 ..... I-23
I.2.35 ASFACT_WC_LIST_DE_0 ..... I-24
I.2.36 ASFACT_WC_LIST_ED_0 ..... I-24
I.2.37 ASFACT_WC_LOGINS_0 ..... I-25
I.2.38 ASFACT_WC_PEOPLEC_0 ..... I-25
I.2.39 ASFACT_WC_PEOPLEC_1 ..... I-26
I.2.40 ASFACT_WC_PEOPLEC_2 ..... I-26
I.2.41 ASFACT_WC_PEOPLEC_3 ..... I-26
I.2.42 ASFACT_WC_PORTLET_0 ..... I-26
I.2.43 ASFACT_WC_SEARCHE_0 ..... I-27
I.3 Analytics Dimension Table Descriptions I-28
I.3.1 ASDIM_WC_APPLICA_0 ..... I-28
I.3.2 ASDIM_WC_CLIENT__0 ..... I-28
I.3.3 ASDIM_WC_DISCUSS_0 ..... I-29
I.3.4 ASDIM_WC_DISCUSS_1 ..... I-29
I.3.5 ASDIM_WC_DISCUSS_2 ..... I-29
I.3.6 ASDIM_WC_DISCUSS_3 ..... I-30
I.3.7 ASDIM_WC_DOCUMENT_0 ..... I-30
I.3.8 ASDIM_WC_EVENTS_0 ..... I-31
I.3.9 ASDIM_WC_GROUPSP_0 ..... I-31
I.3.10 ASDIM_WC_LISTS_0 ..... I-31
I.3.11 ASDIM_WC_PAGES_0 ..... I-32
I.3.12 ASDIM_WC_PORTLET_0 ..... I-32
I.3.13 ASDIM_WC_PORTLET_1 ..... I-32
I.3.14 ASDIM_WC_PRODUCE_0 ..... I-33
I.3.15 ASDIM_WC_REFERRE_0 ..... I-33
I.3.16 ASDIM_WC_SEARCHE_0 ..... I-33
I.3.17 ASDIM_WC_TAGS_0 ..... I-33
I.3.18 ASDIM_WC_USER_AG_0 ..... I-34
I.3.19 ASDIM_TIME I-34
I.3.20 ASDIM_USERS I-35

I.3.21 ASDIM_USERPROPERTIES I-35
I.3.22 ASDIM_USERPROPERTYVALUES I-35
I.4 Analytics System Table Descriptions I-35
I.4.1 ASSYS_CONFIG I-36
I.4.2 ASSYS_DIMENSIONPROPS I-36
I.4.3 ASSYS_EVENTDIMENSIONS I-36
I.4.4 ASSYS_EVENTFACTS I-37
I.4.5 ASSYS_EVENTS I-37
I.4.6 ASSYS_NAMESPACES I-38
I.4.7 ASSYS_REPORTCOMPOSITION I-38
I.4.8 ASSYS_REPORTGROUP I-38
I.4.9 ASSYS_REPORTITEMS I-39
I.4.10 ASSYS_REPORTITEMVALUES I-39
I.4.11 ASSYS_REPORTS I-40
I.5 Analytics User Properties I-42
I.5.1 User Property Tables - Overview I-42
I.5.2 User Property Dimension Tables I-42
I.5.2.1 ASDIM_USERPROPERTIES I-42
I.5.2.2 ASDIM_USERPROPERTYVALUES I-43
I.5.3 Out-of-the-Box User Properties for WebCenter Portal I-43
I.6 Analytics Event and Dimension Data Types I-44

Glossary

Index

Preface

This guide explains how to build portal applications using WebCenter Portal: Framework and provides in-depth information for all of the following tasks:

- How to set up and prepare your development environment for portal development, including iterative and round-trip development.
- How to plan, build, deploy, and manage a Framework application.
- How to secure, administer, monitor, and maintain a Framework application and all of its associated components.
- How to develop custom navigation user interfaces for a portal.
- How to prepare an application for, configure, and integrate Oracle WebCenter Portal services.
- How to integrate content into a Framework application.
- How to deliver personalized content to a Framework application.
- How to plan, build, deploy, and manage portlets for a Framework application.

> **Note:** For the portable document format (PDF) version of this manual, when a URL breaks onto two lines, the full URL data is not sent to the browser when you click it. To get to the correct target of any URL included in the PDF, copy and paste the URL into your browser's address field. In the HTML version of this manual, you can click a link to directly display its target in your browser.

Audience

This guide assumes the audience has already read and performed the steps in the Oracle Fusion Middleware Tutorial for Oracle WebCenter Portal Developers.

This manual is written for all of the following developers:

- The developer who wants to build a WebCenter Portal: Framework application or integrate Oracle WebCenter Portal functionality to their application.
- The component developer, who wants to build portlets from Oracle WebCenter Portal services.

This guide also assumes that the audience has already read the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* and is familiar with the following technologies:

- Java

- Oracle JDeveloper
- JavaServer Faces
- Oracle Application Development Framework (Oracle ADF) (purpose, basic architecture, basic development skills)
- Oracle ADF Faces components
- Oracle WebLogic Server

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Fusion Middleware 11*g* Release 1 (11.1.1.6.2) documentation set or on Oracle Technology Network (OTN) at http://www.oracle.com/technology/index.html.

- *Oracle Fusion Middleware Tutorial for Oracle WebCenter Portal Developers*
- *Oracle Fusion Middleware Tutorial for Oracle WebCenter Portal: Spaces*
- *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- *Oracle Fusion Middleware Error Messages Reference*
- *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New

What's new in Release 11.1.1.6? Faster performance, simplified connection interfaces, and enhanced design-time experience to name a few. Read this quick-reference page for a concise summary of what's new in this release and for pointers to more detailed information.

Note: Release 11.1.1.6.0 is part of a *patch set*. A patch set is a single-installation collection of patches that are designed to be applied together. Patching involves copying a small collection of files over an existing installation.

A patch is normally associated with a particular version of an Oracle product and involves updating from one minor version of the product to a newer minor version of the same product (for example, from Release 11.1.1.5.0 to Release 11.1.1.6.0).

All users must install the Release 11.1.1.6.0 patch set for continued support.

New Features for Release 11.1.1.6

The following table lists and describes new features in this release and provides links to more detailed information in our guides.

Feature	Description	For More Information, See…
Blogs	More special characters allowed in blog names: ■ When you create a blog, you can now use the following special characters in blog names in addition to letters and numbers: ? \ / : [ ] * ' " | Enhanced commenting: ■ Commenting is now available when you view the list of blog entries and not on just the entry itself.	■ "Working with Blogs," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

Feature	Description	For More Information, See…
Certification	More technologies certified against the WebCenter Portal application platform: ▪ Microsoft Lync 2010 for IMP ▪ WebSphere Application Server V7 ▪ Internet Explorer 9 Browser ▪ Chrome Browser 10 and chrome support ▪ Microsoft Office 2010 on Windows XP (32-bit and 64 bit)	▪ "Microsoft Lync Prerequisites," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* ▪ Oracle Fusion Middleware Supported System Configurations page at `http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html`. ▪ "Working with Microsoft Office and Explorer Integration," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Discussions service	▪ Discussions task flows are streamlined for greater ease of use.	▪ "Working with the Discussions Service," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Documents service	▪ Follow a clearer set of steps to connect to the Oracle WebCenter Content repository. ▪ Upload multiple files from your desktop by drag and drop. ▪ Select and upload multiple files from the Document Explorer in one step. ▪ View file upload progress. ▪ Share documents more easily. ▪ Perform in-context workflow approval. Approver can view and approve or reject unreleased content from the contribution mode without needing to enter edit mode. ▪ Enter contribution approval mode on a page with content (either HTML or Site Studio) in a workflow. View and approve or reject unreleased content from contribution approval mode without entering edit mode.	▪ "Managing Content Repositories," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* ▪ "Configuring the Documents Service Content Repository Connection," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* ▪ "Uploading New Files," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ "Working with the Documents Service Task Flows and Document Components," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ "Publishing Content Using Content Presenter," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ "Approving or Rejecting a File in a Workflow," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Events service	▪ Task flows are streamlined to enhance usability. For example, users can now view links to events from both calendar and list views.	▪ "Working with the Events Service," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

Feature	Description	For More Information, See…
Notifications	▪ Subscription messages now use IMIP rather than iCal for notifications about events. Consequently, event notifications are added automatically to your Outlook Calendar, without your having to open the notification attachment and add it yourself.	▪ "Subscribing to Events," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Pagelet Producer	More robust Pagelet Producer: ▪ Use SharePoint web parts, including those requiring Kerberos authentication. ▪ Includes support for logging of HTTP traffic between a proxy and a resource. ▪ Added detailed pagelet debug logging. ▪ Available as a unified producer for non-standards-compliant web applications. ▪ Enables consumption of gadgets from other OpenSocial containers, such as iGoogle, and exposure of them as pagelets that can be consumed by WebCenter Portal sites, other Oracle portals, or Oracle Web Content Management for WebCenter sites. ▪ Enables production of OpenSocial gadgets and exposure of all existing pagelets as gadgets.	▪ Chapter 63, "Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer" ▪ "Managing the Oracle WebCenter Portal's Pagelet Producer," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
Pages	▪ New Administrator-level **Delete Personalization** option enables removal of all users' customizations from business role pages or personal pages in one step. ▪ New system pages are available to support viewing and managing application metrics, space memberships, spaces, and space templates. ▪ The **Subscribe** system page is now named **Self-Service Membership** to more-closely reflect its use case.	▪ "Removing All User Customizations from a Business Role Page," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ "Removing All User Customizations from a Personal Page," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ "Working with System Pages," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Performance	Accelerated performance at design time and runtime: ▪ Experience faster initial page downloads. ▪ Forgo session creation for unauthenticated users. ▪ Realize faster creation times for WebCenter Portal: Framework applications.	—

Feature	Description	For More Information, See…
Portlet producers	▪ New Enterprise 2.0 portlet producer is available for exposing task flows from WebCenter Portal services as portlets. Integrate portlets with WebCenter Portal applications, Oracle WebLogic Portal, and Oracle WebCenter Interaction. The new Enterprise 2.0 portlet producer is certified with WebSphere, MySQL, Internet Explorer 9, and Chrome 10.	▪ Chapter 58, "Overview of Portlets" ▪ Section 64.7, "Consuming WebCenter Services Portlets" ▪ "Configuring the E2.0 Portlet Producer," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
Publisher	▪ Upload multiple files at once through the Publisher task flow.	▪ "Sharing Files Through the Publisher Task Flow," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Search service	▪ There are more task flow parameters to limit the scope of searches. Administrators can narrow the scope of searches to specific spaces, services, and document types. They also can add attributes to the list of standard attributes returned with each search result item, and they can hide standard refiners available to users with search results. ▪ You can now build portals using a navigation model that enables SES and public search engines, such as Google and Bing, to crawl and index the full text content of pages.	▪ "Working with the Search Service," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ▪ Chapter 11, "Building a Navigation Model for Your Portal" ▪ Chapter 45, "Integrating the Search Service" ▪ "Managing Oracle SES Search in WebCenter Portal," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
Security	▪ Support is available for an OAuth2 Provider in an OAuth client. ▪ Support is available for dynamically determining group membership via integration with Oracle Entitlements Server.	▪ Chapter 69, "Securing Your WebCenter Portal: Framework Application" ▪ "Managing WebCenter Portal Application Security," in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* ▪ "Understanding Security," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

Feature	Description	For More Information, See…
Tags service	■ We have updated the edit tags user interface in the Document Explorer task flow. ■ The Tags button component and the Edit Tags task flow have a new parameter that enables developers to disable the Shared control. ■ The previously-named "Tag Cloud" task flow is now called the "Tag Selection" task flow; and a new standalone Tag Cloud task flow is available in the design-time catalog.	■ "Working with the Tags Service," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* ■ Chapter 44, "Integrating the Tags Service"
Wikis	■ Access wiki creation features with greater ease: Use a **New Wiki Document** button in a Documents service task flow or in the wiki itself. ■ Experience improved performance in loading wikis, launching the wiki editor, and saving changes to your wiki.	■ "Creating a Wiki Document Using the New Wiki Document Action," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

Part I

Introduction

Part I contains the following chapters:

- Chapter 1, "Developer's Quick Start Guide"
- Chapter 2, "Understanding Oracle WebCenter Portal"
- Chapter 3, "Preparing Your Development Environment"
- Chapter 4, "Working Productively in Teams"
- Chapter 5, "Creating Framework and Portlet Producer Applications"
- Chapter 6, "Advanced Configuration Options"
- Chapter 7, "Preparing Your Application for WebCenter Portal Services"
- Chapter 8, "Introducing the AviTrust Portal Framework Sample Application"

1 Developer's Quick Start Guide

As a developer, if you are preparing to build a WebCenter Portal: Framework application for the first time, you need a good roadmap that shows you where to start and helps guide your initial experience. This chapter provides guidance, best practices, and tips for Framework application developers setting up their development environment for the first time.

> **Tip:** If you are just getting started, we recommend that you look at Section 1.1, "WebCenter Portal Developer's FAQ" first.

This chapter includes the following sections:

- Section 1.1, "WebCenter Portal Developer's FAQ"
- Section 1.2, "Installing Required Software"
- Section 1.3, "Preparing for Team Development"
- Section 1.4, "Creating a New Framework Application"
- Section 1.5, "Preparing for Iterative Development"
- Section 1.6, "Creating Application Resource Connections"
- Section 1.7, "Planning the Portal Layout"
- Section 1.8, "Working with Skins"
- Section 1.9, "Preparing for Round-Trip Development"
- Section 1.10, "What Configuration Files Should I Know About?"
- Section 1.11, "Installing Database Schemas"

1.1 WebCenter Portal Developer's FAQ

Table 1–1 addresses basic questions to help you get started with Framework application development.

> **Tip:** Before you begin to design and build a Framework application, we recommend that you read Chapter 9, "Understanding Framework Applications." This chapter presents an overview of Oracle WebCenter Portal: Framework, and provides tips and links to more detailed information.

Table 1–1 WebCenter Portal Developer's FAQ

Question	Answer
What software do I need to install before getting started?	At a minimum, you need Oracle JDeveloper and the WebCenter Portal extension for Oracle JDeveloper (a JDeveloper add-in that includes WebCenter Portal functionality). Typical portal environments also include database, content repository, and search engine installations. See Section 1.2, "Installing Required Software."
How do I set up source control for my team?	JDeveloper can integrate with a variety of source control systems. See Section 1.3, "Preparing for Team Development."
What can I do to increase my team's productivity?	The best way to improve your team's productivity is by enabling **iterative development**. With iterative development enabled, you can see portal file changes updated in runtime by simply refreshing the browser. See Section 1.5, "Preparing for Iterative Development."
How do I create a new Framework application?	WebCenter Portal provides an out-of-the-box template that creates a pre-configured Framework application for you. This basic portal application includes all of the features you need to get started including a default set of pages, out-of-the-box page templates, and navigation. See Section 1.4, "Creating a New Framework Application."
The default Framework application includes files and features I do not intend to use. Can I remove them?	Yes, you can add and remove Framework application components. For more information, see Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."
How do I make database and content repository connections?	JDeveloper provides a connection wizard that lets you create these kinds of connections. These connections seamlessly integrate with certain components that you may use in a portal, like the Content Presenter task flow. See Chapter 27, "Managing Content Repository Connections."
I'd like to create a highly customized navigation UI for my portal. Is that possible?	Yes, you can control all aspects of how the navigation UI is rendered using the APIs provided by WebCenter Portal: Framework. A popular technique is to use EL to customize how the navigation UI is rendered. For example, you can hide certain links based on the user's role or change the look and feel of a link based on specified criteria. For more information, see Chapter 14, "Visualizing Your Portal Navigation."
How is portal security configured by default?	By default, a Framework application is configured with ADF security. A default username and password (weblogic/weblogic1) are created automatically for you, and you can use this username/password combination for testing purposes immediately. See also Chapter 69, "Securing Your WebCenter Portal: Framework Application."
How do I customize the way content is displayed in my portal?	One common technique is to use the Content Presenter task flow. Content Presenter lets you add content to a page and customize the way the content is displayed using templates. For more information, see Chapter 26, "Introduction to Integrating and Publishing Content."
How do I create a consistent layout for the pages in my portal?	The page template feature allows you to create a layout that any page in your portal can share. In addition to the basic page layout, you can also add features like navigation user interfaces to the template. See Section 1.7, "Planning the Portal Layout."
There are so many ADF components. How do I figure out which ones to use?	WebCenter Portal: Framework provides the WebCenter Portal – Framework Catalog. This catalog includes a collection of components commonly used for building portals. For more information, see Chapter 16, "Creating and Managing Resource Catalogs."
I'd like to change the default skin. How do I do that?	A skin is a style sheet that defines the look of your portal. WebCenter Portal: Framework provides a portal skin that you can conveniently edit in JDeveloper. For more information, see Section 1.8, "Working with Skins."

Table 1–1 (Cont.) WebCenter Portal Developer's FAQ

Question	Answer
Can developers modify or add new portal features that have been deployed into production?	WebCenter Portal provides a solution for round-trip development. See Section 1.9, "Preparing for Round-Trip Development."
What do I need to know about moving the portal from the development environment to the testing, staging, and production environments?	The portal life cycle describes the path a portal takes from development and testing environments to a staging environment and, ultimately, a production environment. For detailed information on managing the life cycle, see Chapter 10, "Understanding the WebCenter Portal Life Cycle."
Where can I get more background information on developing Framework application?	We recommend that you review Chapter 9, "Understanding Framework Applications." This chapter presents an overview of Oracle WebCenter Portal: Framework, and provides tips and links to more detailed information.

1.2 Installing Required Software

A typical development environment includes these components:

- **JDeveloper** – See Section 3.1, "Installing Oracle JDeveloper."
- **WebCenter Portal's Extension for Oracle JDeveloper** – See Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper."
- **Supported Database** – (Optional) For example Oracle Database 11g.
- **Supported Content Repository** – (Optional) For example, Oracle WebCenter Content 11g.
- **Search Engine** – (Optional) For example, Oracle Secure Enterprise Search (SES).

See also the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

1.3 Preparing for Team Development

It's important to consider the overall **team development environment**. To be useful, a team development environment must be configured to allow developers to share common resources, like databases, content repositories, and source code. A well-planned team environment allows you to quickly and consistently develop, build, and update your WebCenter Portal: Framework applications.

> **Tip:** Typically, one member of the development team creates a new Framework application and checks it in to the source code repository. You can create any required database or content repository connections and check those in as well.

For more detailed information on team development topics like source control and file sharing, see Chapter 4, "Working Productively in Teams."

1.4 Creating a New Framework Application

WebCenter Portal: Framework provides a wizard for creating new portal applications. The wizard uses an out-of-the-box **WebCenter Portal – Framework Application** template that ensures the appropriate application components are included.

After the Framework application is created, you can then configure database and content repository connections and check those in to the source control system as well.

If you wish, you can manually configure the Framework application. For example, you might decide to remove some components from the framework application if you know you will never need them.

For detailed information on creating, extending, and modifying Framework applications, see Chapter 5, "Creating Framework and Portlet Producer Applications." For more information on managing connections, see Section 1.6, "Creating Application Resource Connections."

1.5 Preparing for Iterative Development

This section discusses WebCenter's iterative development feature.

- Section 1.5.1, "What Is Iterative Development?"
- Section 1.5.3, "Enabling Iterative Development"
- Section 1.5.4, "Turning Off Iterative Development"

1.5.1 What Is Iterative Development?

Iterative development lets you make changes to your Framework application while it is running on the Integrated WebLogic Server and immediately see the effect of those changes simply by refreshing the current page in your browser. The iterative development feature works by disabling certain optimization features. Iterative development allows developers to work more quickly and efficiently when building a Framework application.

For example, iterative development lets you see changes to these components almost instantly upon a browser refresh:

- page definitions
- navigation model
- page hierarchy
- existing JSPX files
- page templates
- resource catalog
- addition of task flows to pages
- addition of portlets to pages

The following kinds of operations are not supported by iterative development. These operations require you to re-run the application:

- creating any *new* file explicitly (JSPX, page definition, page hierarchy, and so on)
- creating a new file implicitly. For instance, when you add a sub-page to a node in the page hierarchy, a new `*pages.xml` file is created.
- editing any configuration file, like `web.xml` or `adfc-config.xml`.

1.5.2 How Does Iterative Development Work?

Iterative development works by turning off certain MDS and runtime layer caches. Because these performance optimization features are disabled, you might notice some slower performance when running your application in a development environment.

Note: When an application is deployed to the Integrated WebLogic Server, the `org.apache.myfaces.trinidad.CHECK_FILE_MODIFICATION` flag is automatically set to `true` in the `web.xml` file on the server. This setting causes the server to automatically check the modification time of your JSP and skinning CSS files, and discard saved state when they change. This configuration occurs whether or not the iterative development feature is enabled.

When iterative development is enabled, the following changes to application configuration occur:

- The MDS Cache size is set to `0`. This setting causes all metadata files to be re-loaded on each request.
- The Navigation Model cache is invalidated on each request.

For skin development, you can set the application to use uncompressed skins. This setting is not the default and should only be used in a development environment. You can update `web.xml` to enable this functionality with the following context parameter:

```
<context-param>
    <param-name>org.apache.myfaces.trinidadinternal.DISABLE_CONTENT_COMPRESSION</param-name>
    <param-value>true</param-value>
</context-param>
```

Note: When consuming Oracle JSF Portlet Bridge Portlets in your application, ADF attempts to share the skin between the consumer and producer. Therefore, if you are disabling compression on the consumer, you should also do that on the producer. Otherwise, the producer will not generate the correct code to match the uncompressed IDs generated by the consumer. See also Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge."

Note that if you edit the base definition (source) of a resource at runtime while the portal is running in the Integrated WebLogic Server, iterative development will not work as expected: if you edit the resource file in JDeveloper and save, you will not see the change when the running page is refreshed. The reason is because when a resource is edited at runtime at the base document level in the Integrated WebLogic Server, a *copy* of the resource's base document is created in the MDS write directory. From that point forward, this copied version will be used, and the version of the base document in JDeveloper will be ignored. If you run into this issue and want to pick up changes made to the file in JDeveloper, you will need to re-run the application after taking the following steps:

1. In the Application Navigator, click the **Application Menu** icon next to your application's name and choose **Application Properties**.
2. In the Application Properties dialog, expand the **Run** node in the left pane, then select **MDS**.
3. In the right pane, select **Delete customizations before each run**. Doing so clears the MDS of any runtime customizations every time the application is run.
4. Click **OK**.

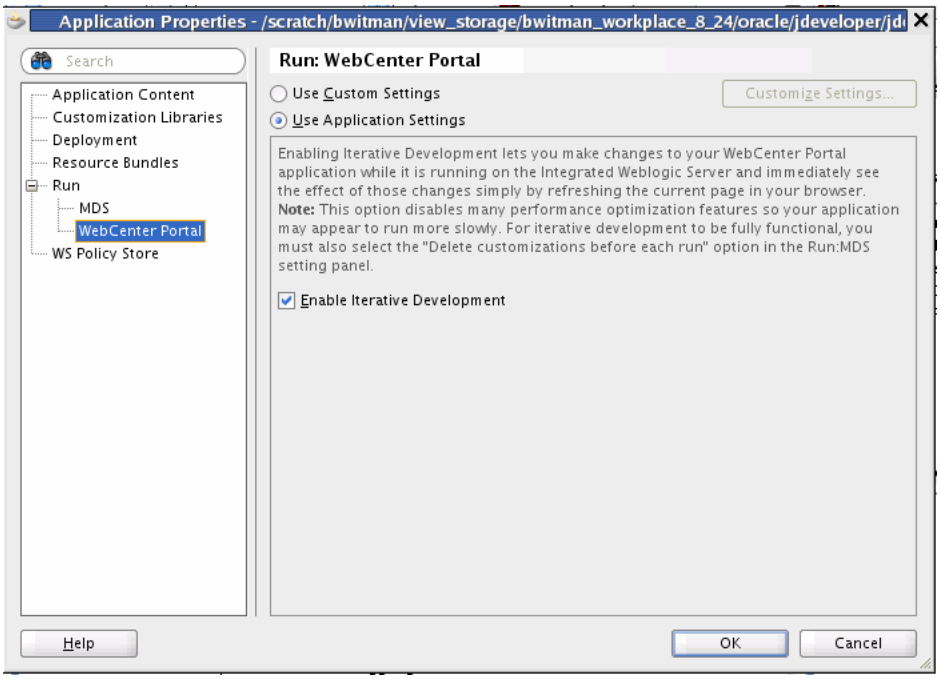
1.5.3 Enabling Iterative Development

This section explains how to enable iterative development, if it has been disabled.

> **Note:** The iterative development feature is enabled by default when you create a new Framework application.

1. Select **Application Properties** from the Application menu.
2. Along the left side of the Application Properties dialog, expand the **Run** node.
3. Select **WebCenter Portal**.
4. Select **Enable Iterative Development**, as shown in Figure 1–1.

Figure 1–1 Enabling Iterative Development



5. Click **OK**.

1.5.4 Turning Off Iterative Development

This section explains how to turn off iterative development. Iterative development is enabled by default; however, it's a good practice to turn off iterative development for testing purposes before you deploy your application to a production server.

1. Select **Application Properties** from the Application menu.
2. Along the left side of the Application Properties dialog, expand the **Run** node.
3. Select **WebCenter Portal**.
4. Unselect **Enable Iterative Development**.
5. Click **OK**.

1.6 Creating Application Resource Connections

Connections allow the Framework application to access external data and services. For example, if you wish to use the Content Presenter task flow to display content from an Oracle WebCenter Content Server repository, you need to configure a connection to the repository. If you intend to consume portlets from a portlet producer, you need to configure the producer connection.

> **Tip:** A good practice is to create and test your connections once and check them into your source control system. Then, other developers on your team can check out the connections and use them. This technique also allows your team to keep in sync whenever a connection changes.

This section describes the different ways to access the wizards for creating new connections. See also Section 1.6.3, "Where Can I Learn More About Connections?."

1.6.1 Where Are Connections Located?

Depending on how you invoke a wizard to create a connection, connections are placed in one of the following locations:

- Under Application Resources in the Application Navigator

  Connections created here can be used in the current application only. This is the most common way to create a repository connection.

  For certain services, you can drag and drop a connection from Application Resources onto a page to create different types of task flow regions. To learn more, refer to the individual WebCenter Portal: Services chapters.

- Under IDE Connections in the Resource Palette

  Connections created here can be reused across Framework applications. To use these connections in an application, you just have to drag and drop the connection from the Resource Palette onto the Connections node in that application.

1.6.2 How Do I Access the Connection Wizards?

To access a connection wizard from the New Gallery:

1. From the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **Connections**, select the type of connection you want to create, and click **OK**.

   Depending on your selection, the **Create <*Connection_Type*> Connection** dialog opens.
3. The **Create Connection in** option is set to **Application Resources** by default.

   You can select **IDE Connections** to create a connection in the Resource Palette.

To access a connection wizard from the Application Navigator:

1. Right-click the **Connections** node under Application Resources and select **New Connection**, then select the connection type from the context menu.
2. Depending on your selection, the **Create <*Connection_Type*> Connection** dialog or wizard opens.

   The **Create Connection in** option is set to **Application Resources** by default.

To access a connection wizard from the Resource Palette:

1. Click the **New** icon in the Resource Palette, select **New Connection**, then select the connection type from the context menu.
2. Depending on your selection, the **Create <*Connection_Type*> Connection** dialog or wizard opens.

   The **Create Connection in** option is set to **IDE Connections** by default.

1.6.3 Where Can I Learn More About Connections?

For information about creating and consuming connections, see:

- Chapter 27, "Managing Content Repository Connections"
- Chapter 64, "Consuming Portlets"
- Part IV, "Integrating and Publishing Content"
- Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application"

1.7 Planning the Portal Layout

You can use **page templates** to control the layout of your portal. A page template is a JSPX file that specifies the look and feel of your portal's pages. The template defines header, footer, content, and navigation regions within the page. You can apply the template to any number of pages, resulting in a consistent look and feel.

WebCenter Portal: Framework provides several default page templates to help you get started. For more information, see Chapter 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

1.8 Working with Skins

A **skin** is a CSS file that defines the overall look of a portal, including color schemes, fonts, and the display characteristics of ADF Faces components. WebCenter Portal: Framework also provides a portal skin file that you can conveniently edit in JDeveloper. This file lets you override settings in the default skin. Just open the file, uncomment any components you wish to modify, and save the file when you are finished making changes. For details, see Chapter 15, "Creating and Managing Skins."

1.9 Preparing for Round-Trip Development

Round-trip development refers to features and techniques that allow you to retrieve resources from a deployed, runtime portal back to JDeveloper for maintenance or enhancement. After modifying a resource in JDeveloper, you can use the Resource Manager to upload the resource back to the deployed portal. WebCenter Portal's round-trip development features provide a simple, convenient way to modify portal resources without redeploying the entire application.

For more information on round-trip development, see Section 17.3, "Enabling Runtime Administration of Your Portal" and Section 17.4, "Enabling Round-Trip Development of Resources."

1.10 What Configuration Files Should I Know About?

A number of automatically generated configuration files are placed under the `WEB-INF` and `Page Flows` folders of a portal project when you create a Framework application.

> **Tip:** For a complete listing of portal configuration files, see Appendix H, "Manually Migrating a Framework Application."

This section briefly describes these file:

- Section 1.10.1, "The Framework Application Template Default web.xml File"
- Section 1.10.2, "The Framework Application Template Default faces-config.xml File"
- Section 1.10.3, "The Framework Application Template Default trinidad-config.xml File"
- Section 1.10.4, "The Framework Application Template Default adfc-config.xml File"

1.10.1 The Framework Application Template Default web.xml File

Part of Framework application configuration is determined by the content of its J2EE application deployment descriptor file: `web.xml`. The `web.xml` file defines many application settings that are required by the application server.

> **Note:** Rather than being specified in `web.xml`, the context root path is assigned when the application is deployed.

Typical runtime settings include initialization parameters, custom tag library location, and security settings. Depending on the technology you use, other settings may be added to `web.xml` as appropriate.

> **Note:** For standard Java EE files like `web.xml`, there is usually a counterpart Oracle-specific file with additional Oracle-specific options, for example, `weblogic.xml`.

The `web.xml` file is located in the `/public_html/WEB-INF` directory relative to the project in your Framework application.

When you first create your Framework application, configuration settings for JSF servlet and mapping and for resource servlet and mapping are automatically added to the starter `web.xml` file.

The `Faces Servlet` entry inside `<servlet></servlet>` tags provides information about the JSF servlet, `javax.faces.webapp.FacesServlet`. This servlet manages the request processing life cycle for web applications that use JSF to construct the user interface. The configuration setting maps the JSF servlet to a symbolic name, *Faces Servlet*.

The `resources` entry inside `<servlet></servlet>` tags provides information about the ADF resource servlet used to serve up web application resources (images, style sheets, JavaScript libraries) by delegating to a `ResourceLoader`.

The <servlet-mapping></servlet-mapping> tags map the URL pattern to a servlet's symbolic name. You can use either a path prefix or an extension suffix pattern.

By default, JDeveloper uses the path prefix /faces/*. That is, when a URL, for example, http://localhost:8080/SRDemo/faces/index.jsp, is issued, the URL activates the JSF servlet, which strips off the faces prefix and loads the file /SRDemo/index.jsp.

Note: If you prefer to use the extension jsf for web pages instead of jsp or jspx, then you must set the javax.faces.DEFAULT_SUFFIX context parameter to jsf, for example:

```
<context-param>
   <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
   <param-value>.jsf</param-value>
</context-param>
```

Then add a servlet mapping in web.xml that invokes the JSP servlet for files with the extension jsf.

To edit web.xml in Oracle JDeveloper, right-click web.xml in the Application Navigator and choose **Open** from the context menu. This opens web.xml in the Web Application Deployment Descriptor editor, in Overview mode. If you are familiar with configuration element names, then you can also use the XML editor to modify web.xml.

For information about the configuration elements you can use in the web.xml file, see "Oracle ADF XML Files" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

1.10.2 The Framework Application Template Default faces-config.xml File

Use the faces-config.xml file to register a Framework application's resources, such as custom validators and managed beans, and to define all page-to-page navigation rules. The default name of this file is faces-config.xml, though this naming convention is not required.

Depending on how resources are packaged, an application can have one or multiple faces-config.xml files. For example, you can create individual JSF configuration files for:

- Different areas of your application
- Each library containing custom components or renderers

For more information about creating multiple faces-config.xml files, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

Example 1–1 illustrates the default faces-config.xml file provided through the WebCenter Portal: Framework Application template. This file is located in the /public_html/WEB-INF directory relative to the project in your Framework application.

Example 1–1 Default faces-config.xml File Provided Through the Framework Application Template

```
<?xml version="1.0" encoding="US-ASCII"?>
<faces-config version="1.2" xmlns="http://java.sun.com/xml/ns/javaee">
  <application>
```

```
    <default-render-kit-id>oracle.adf.rich</default-render-kit-id>
    <view-handler>oracle.webcenter.portalframework.sitestructure.handler.CustomViewHandler
    </view-handler>
  </application>
  <managed-bean>
    <managed-bean-name>preferenceBean</managed-bean-name>
    <managed-bean-class>oracle.webcenter.portalframework.sitestructure.PreferencesBean
    </managed-bean-class>
    <managed-bean-scope>application</managed-bean-scope>
  </managed-bean>
</faces-config>
```

To edit the `faces-config.xml` file, double-click it in the Application Navigator. By default, the file is opened in the Editor window in *Diagram* mode, as indicated by the active **Diagram** tab at the bottom of the Editor window. When creating or modifying JSF navigation rules, Diagram mode enables you to visually create and manage page flows. For more information, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

To create or modify configuration elements other than navigation rules, use the Editor's Overview mode. Enter this mode by clicking the **Overview** tab at the bottom of the Editor window.

JSF allows multiple `<application>` elements in a single `faces-config.xml` file. When you use the JSF Configuration Editor, you can edit only the first instance. For any other `<application>` elements, you must edit the file directly using the XML editor. To use the XML editor, open the `faces-config.xml` file and go to the **Source** tab in the Editor window.

For reference information about the configuration elements you can use in the `faces-config.xml` file, see "ADF Faces Configuration" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

1.10.3 The Framework Application Template Default trinidad-config.xml File

The default `trinidad-config.xml` file, available when you create a Framework application, contains information about the application skin family. Additionally, it can contain information about the level of page accessibility support, page animation, time zone, enhanced debugging output, and the URL for Oracle Help for the Web (OHW). Like `faces-config.xml`, the `trinidad-config.xml` file has a simple XML structure that enables you to define element properties using JSF Expression Language (EL) or static values.

Example 1–2 illustrates the default `trinidad-config.xml` file provided through the WebCenter Portal: Framework Application template. This file is located in the `/public_html/WEB-INF` directory relative to the project in your Framework application.

Example 1–2 Default trinidad-config.xml File Provided Through the Framework Application Template

```
<?xml version="1.0" encoding="US-ASCII"?>
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">

  <skin-family>fusion</skin-family>

</trinidad-config>
```

In addition to the skin family, you can define the following application values in the `trinidad-config.xml` file:

- Page animation
- Level of page accessibility support
- Time zone
- Enhanced debugging output
- Oracle Help for the Web (OHW) URL

For reference information about the `trinidad-config.xml` file, see "ADF Faces Configuration" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

1.10.4 The Framework Application Template Default adfc-config.xml File

The `adfc-config.xml` file is the configuration file for an ADF unbounded task flow. This file contains metadata about the activities and control flows contained in the unbounded task flow. The default name for this file is `adfc-config.xml`, though this naming convention is not required.

If **ADF Page Flow** is specified as a Selected Technology on the Technology Scope page of the Project Properties dialog, a new `adfc-config.xml` source file is automatically created within the project. The `adfc-config.xml` file is the main source file for an unbounded task flow.

The `adfc-config.xml` file is located in the `/public_html/WEB-INF` directory relative to the project in your Framework application.

Note: If you do not plan to use task flows in your Framework application, you can delete the `adfc-config.xml` file.

For more information about task flows and the `adfc-config.xml` file, see "Oracle ADF XML Files" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

1.11 Installing Database Schemas

Certain Oracle WebCenter Portal components require schemas that must be installed in a supported database. For details, see Section 3.5, "Installing Database Schemas."

2

Understanding Oracle WebCenter Portal

This chapter introduces you to Oracle WebCenter Portal and helps you understand what you must consider when using WebCenter technology to build an application.

This chapter includes the following sections:

- Section 2.1, "What Is Oracle WebCenter Portal?"
- Section 2.2, "Introducing the WebCenter Portal: Framework"
- Section 2.3, "What Are the Main WebCenter Portal: Framework Features?"
- Section 2.4, "Introducing Oracle WebCenter Portal: Services"
- Section 2.5, "Introducing Oracle Application Development Framework (Oracle ADF)"
- Section 2.6, "Introducing Oracle JDeveloper and the WebCenter Portal Extension"
- Section 2.7, "Understanding the Application Life Cycle"
- Section 2.8, "Design Questions to Consider Before You Start"
- Section 2.9, "Overview of WebCenter Accessibility Features"

Audience

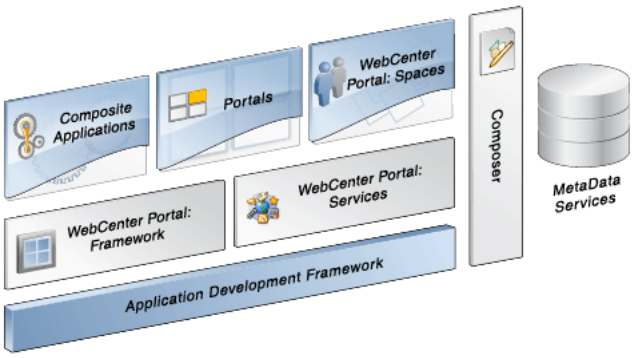
This chapter is intended for developers of Framework applications. Developers should review this chapter carefully to determine what options are available to them.

2.1 What Is Oracle WebCenter Portal?

Oracle WebCenter Portal provides design time and runtime tools for building enterprise portals, transactional web sites, and social networking sites. Typical WebCenter use cases include employee intranets, customer extranets, dashboards, blogs and wikis, community discussion forums, online stores, and many others. WebCenter provides the tools you'll need to create these kinds of applications.

Figure 2–1 provides an overview of the Oracle WebCenter Portal architecture, showing the major components that comprise the product.

Figure 2–1 Overview of Oracle WebCenter Portal Architecture



2.2 Introducing the WebCenter Portal: Framework

The Oracle WebCenter Portal: Framework provides portal-specific features to a Framework application. These features include page hierarchies, navigation models, delegated security, customization, and others. Portals can also include features like portlets, content management system integration, personalization, social computing services, search, analytics, and more.

From the perspective of an end user, a portal is a web site with pages that are organized by tabs or some other form of navigation. Each page can contain portlets, static content, dynamic content, and task flows. Authorized users can also customize their view of a portal by adding their own pages, adding portlets of their choosing, and changing the Look And Feel of the interface.

For a complete overview of the Oracle WebCenter Portal: Framework, Chapter 9, "Understanding Framework Applications."

2.3 What Are the Main WebCenter Portal: Framework Features?

This section discusses a few of the major features that can be used in a Framework application. These features support integration of remote applications, page customization, content integration, personalization, security, and others.

- Section 2.3.1, "Portlets and Pagelets for Accessing Remote Applications and Services"
- Section 2.3.2, "Oracle WebCenter Portal's Composer Allows User Customization"
- Section 2.3.3, "Content Integration with Supported Content Repositories"
- Section 2.3.4, "Search Framework"
- Section 2.3.5, "Resource Catalogs Help Organize the Development Experience"
- Section 2.3.6, "Application Security"

2.3.1 Portlets and Pagelets for Accessing Remote Applications and Services

This section discusses two technologies–portlets and pagelets–that let you integrate remote applications into your portal pages.

- Section 2.3.1.1, "What Are Portlets?"
- Section 2.3.1.2, "What Are Pagelets?"

2.3.1.1 What Are Portlets?

A portlet is a reusable Web component that can draw content from many different sources. Portlets provide a means of presenting data from multiple sources in a meaningful and related way. Portlets can display excerpts of other web sites, generate summaries of key information, perform searches, and access assembled collections of information from a variety of data sources. Because different portlets can be placed on a common page, the user receives a single-source experience. In reality, the content may be derived from multiple sources.

For information about the different ways you might create portlets and how you might use portlets, including the pre-built portlets available with WebCenter Portal, see Chapter 58, "Overview of Portlets." For information about consuming portlets on pages and linking them, see Chapter 64, "Consuming Portlets."

2.3.1.2 What Are Pagelets?

A pagelet is a reusable user interface component similar to a portlet, but while portlets were designed specifically for portals, pagelets can be run on any web page, including within a portal or other web application. Pagelets can be used to expose platform-specific portlets in other web environments.Any HTML fragment can be a pagelet, but pagelet developers can also take advantage of many of the features available to portlet developers to write pagelets that are parameterized and configurable, to dynamically interact with other pagelets, and respond to user input using Ajax patterns.

Oracle WebCenter Portal's Pagelet Producer provides a collection of useful tools and features that facilitate dynamic pagelet development. For detailed information about pagelet development and configuration using the Pagelet Producer, see Chapter 63, "Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer."

2.3.1.3 What is the Difference Between Portlets and Task Flows?

As explained in Section 2.3.1.1, "What Are Portlets?," portlets are reusable Web components that can draw content from many different sources. Portlets can manage and display anything from static HTML to Java controls to complex web services and process-heavy applications.

A single portlet can also have multiple instances—in other words, it can appear on a variety of different pages within a single portal, or even across multiple portals if the portlet is enabled for Web Services for Remote Portlets (WSRP). You can customize portlets to meet the needs of specific users or groups.

Task flows let you build customizable applications with reusable units of business logic. They represent a modular approach for defining control flow in an application. Each task flow represents a reusable piece of business logic. With isolated memory and transaction scopes, task flows are decoupled from the surrounding application. This decoupling allows task flows to be included in Oracle WebCenter Portal's Composer, for instance.

A task flow encapsulates control flow rules, activities, and managed beans to implement business modules. Task flows are easily reusable within WebCenter Portal applications. You can drag and drop task flows onto any customizable WebLogic Portal application page. In this sense, you can think of a task flow as a "local portlet" that can be reused and dropped into any ADF application.

While task flows are high-level implementations of business processes, they do not host data-access or business logic. Task flows must interact with data controls and declarative bindings for these purposes. These data controls and bindings in turn interact with the ADF Business Components framework.

Following the Oracle-submitted standard JSR 329, you can expose your task flows as standards based portlets. In this way, one application encompasses both the base application functionality and the portlets to be consumed for integration. When you revise your application, the portlets are naturally and automatically updated right along with it, rather than requiring a separate development project. To support JSR 329, the Oracle WebCenter Portal: Framework provides the Oracle JSF Portlet Bridge. For more information about the Oracle JSF Portlet Bridge, see Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge."

In addition to consuming task flows as portlets, you can consume task flows as shared libraries in a JSF application to enable you to embed these JSF fragments directly. You can wrap the transactions around the task flows along with the other functionality in their application. For more information, see Chapter 14, Getting Started with ADF Task Flows" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

2.3.2 Oracle WebCenter Portal's Composer Allows User Customization

Composer is an easy, browser-based environment that you can add to existing JSF applications. End users can then use Composer to edit the page at runtime to create their own mashups. To make pages editable at runtime in this way, you simply add Composer design time components to a page in JDeveloper. If you create a blank page in JDeveloper with just Composer components, users have the ability to redesign the page while in Edit mode using Composer. Some of the tasks they can perform include:

- Editing page settings and parameters
- Adding content to the page
- Editing component properties and parameters
- Wiring components to page parameters
- Arranging content on the page
- Changing the layout for components on the page
- Removing components from the page

The Composer tag library provides the components that you can add to make a page editable at runtime and define the behavior of content on the page (for example, move, sequence, or hide components).

In Composer, the Component Catalog dialog box, which is displayed by clicking the Add button on the page, displays the default runtime Resource Catalog Viewer. Users can browse the components in the viewer and then add them to the page. By default, the viewer displays the default Resource Catalog, which contains all of the Oracle ADF components and portlets available to the application. To control what components are visible to users in the viewer, you can modify the default Resource Catalog, or create one or more of your own Resource Catalogs. Resource Catalogs can contain items such as portlets, layout components, task flows, documents, and Oracle ADF Faces components.

For more information about implementing Composer components in your application, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."

2.3.3 Content Integration with Supported Content Repositories

JCR (Java Content Repository API, also known as JSR 170) adapters enable you to make data stored in content management systems, such as Oracle WebCenter Content, Oracle Portal, or even your file system, available to your application.

Using JDeveloper, you can use the prebuilt JCR data control to grab the content and drop it onto your page. You can leverage the provided prebuilt user interface to display the content in your Framework application. This architecture enables you to build your user interface once and then at deployment time or during runtime, switch to whatever back end is required without having to recompile or rebuild the application. In addition, you do not need to learn the intricacies of each content management system's custom APIs. For example, you could create a data control that selects content from any JCR 1.0 compliant repository or file system. When the data control is created, you can drop it onto a JSP document as a table.

If you retrieve data from a content repository other than Oracle Portal, Oracle WebCenter Content, or the file system, then you can create your own JCR adapter. From the Content Repository Configuration page of the Create Data Control wizard, you can choose the content repository from which you want to retrieve data.

The Content Presenter task flow lets you conveniently add content to your application. Content Presenter allows you to select a single item of content, multiple content items, or query for content, and then select a template to render the content on a page in a Framework application. In addition, you can design custom display templates for Content Presenter. See Chapter 29, "Creating Content Presenter Display Templates." For more information, see also "Adding Content Using Content Presenter" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* and Chapter 27, "Managing Content Repository Connections."

2.3.4 Search Framework

Search is one of the most common and useful features of any web application. Oracle WebCenter Portal: Framework has a unified, extensible Search framework that enables the discovery of information and people through an intuitive user interface. The powerful Search framework enables you to seamlessly include enterprise-wide search capabilities into your application. With all relevant and secure information easily navigable, users do not need to switch between applications performing multiple searches. See Chapter 45, "Integrating the Search Service."

2.3.5 Resource Catalogs Help Organize the Development Experience

The Resource Catalog provides a federated view of the contents of one or more otherwise unrelated repositories within a unified search and browse user interface. Resources are created and published in their source repository, then exposed to the developer through the JDeveloper Resource Palette and to the end user through the Resource Catalog Viewer. The Resource Palette contains the WebCenter Portal Catalog, which includes elements that are commonly used in portal development.

Resource Catalogs can contain the following components:

- **Layout components.** The primary layout component is a `Box`, which is a container that can hold all other types of components. At runtime, you must have a `Box` into which you can drag and drop components. You can also add and arrange child components and delete components from a `Box`. See the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.
- **Oracle ADF Faces components.** You can add `Text`, `Image`, `Page Link`, `Web Page`, and `Website Link` components to your page. These are analogous to the JDeveloper design time components `Rich Text Editor`, `Image`, `Command Link`, `Web Page`, and `Go Link`, respectively.

The `Text` component enables you to add rich text on the page. Adding this component invokes a text editor that can be resized and is similar to an HTML editor. You can add text and format it using the options available in the editor. See the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

- **Portlets.** You can add WSRP, JSF Portlet Bridge Portlets, or PDK-Java portlets from any producer that was registered in JDeveloper. For more information about registering portlet producers and adding portlets to a page, see Chapter 64, "Consuming Portlets."
- **Pagelets.** Oracle WebCenter Portal's Pagelet Producer provides a collection of useful tools and features that facilitate dynamic pagelet development. For detailed information about pagelet development and configuration using the Pagelet Producer, see Chapter 63, "Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer."
- **Task Flows.** If you have created task flows in JDeveloper, you can add these task flows from JAR files onto your page at runtime. For more information about ADF task flows, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.
- **Documents.** If you have configured the Documents service in your application, you can add documents from this service. For more information about the Documents service, see Chapter 30, "Integrating the Documents Service."
- **Data Controls.** Several WebCenter Portal services provide data controls for building a customized user interface with a Framework application or task flow. The documentation for each service includes more information about that service's data control. See the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

2.3.6 Application Security

With the Oracle ADF extensions provided in Oracle WebCenter Portal, you can define security for an entire application, a page hierarchy, part of a page hierarchy, navigational components, or a single page. You can define security for individual actions provided by customizable components. In addition, for Oracle WebCenter Portal: Services that connect to back-end servers using web services, you can provide secure identity propagation with WS-Security.

Because Oracle WebCenter Portal security is based on the JAAS and J2EE standards, enterprise roles defined in the existing identity management store can be leveraged directly when securing a Framework application. You need not synchronize roles within the application being built. It just references and uses the defined users and roles directly. Note also that you can use file-based security for the development phase of the application and then easily switch over to enterprise identity management at deployment time.

Oracle WebCenter Portal also provides application roles that you can use to represent the policy of an application. By associating permissions with an application role defined within the policy store, you can keep them self contained within the application. On deployment, you can then associate users and enterprise roles with the application roles to grant those permissions to end users.

In some cases, it is desirable to leverage existing applications that have their own authentication mechanism, such as email. The email system is often on a different authentication system (with different user names and passwords) than the new application. The application must map the email user to the application user such that

end users do not have to enter their user names and passwords each time they need information. Oracle WebCenter Portal: Framework provides the means to securely manage these user names and passwords with the External Application functionality.

For more information about options for securing your Framework application, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."

2.4 Introducing Oracle WebCenter Portal: Services

Oracle WebCenter Portal: Services expose social networking and personal productivity features through services, which, in turn, expose subsets of their features and functionality through task flows.

Table 2–1 describes each of the available services.

Table 2–1 Oracle WebCenter Portal: Services

Service	Description	Chapter
Activity Graph	Leverages collective intelligence to benefit search and social applications.	Chapter 46, "Integrating the Activity Graph Service"
Analytics	Enables you to display usage and performance metrics for your Framework application.	Chapter 47, "Integrating the Analytics Service"
Announcements	Provides the ability to post announcements about important activities and events to all authenticated users.	Chapter 33, "Integrating the Announcements Service"
Blog	Provides easy integration of a blog application within the context of your application.	Chapter 32, "Integrating Wikis and Blogs"
Discussions	Provides the ability to create threaded discussions, posing and responding to questions and searching for answers. Also provides an effective group communication mechanism for important activities and events.	Chapter 34, "Integrating the Discussions Service"
Documents	Provides content management and storage capabilities, including content upload, file and folder creation and management, file check out, versioning, and so on.	Chapter 30, "Integrating the Documents Service"
Events	Provides the ability to create and maintain a schedule of events relevant to a wider group of users. Events are published to all authenticated users.	"Working with the Events Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* Chapter 52, "Integrating the Events Service"
Instant Messaging and Presence (IMP)	Provides the ability to observe the status of other authenticated users (whether online, offline, busy, or away) and to contact them instantly.	Chapter 35, "Integrating the Instant Messaging and Presence Service"
Links	Provides the ability to view, access, and associate related information; for example you can link to a solution document from a discussion thread.	Chapter 43, "Integrating the Links Service"

Table 2–1 (Cont.) Oracle WebCenter Portal: Services

Service	Description	Chapter
Lists	Provides the ability to create, publish, and manage lists. Users can create lists from prebuilt structures or create their own custom lists.	"Working with the Lists Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* Chapter 48, "Integrating the Lists Service"
Mail	Provides easy integration with IMAP and SMTP mail servers to enable users to perform simple mail functions such as reading messages, creating messages with attachments, replying to or forwarding existing messages, and deleting messages.	Chapter 36, "Integrating the Mail Service"
Notes	Provides the ability to "jot down" and retain quick bits of personally relevant information. **Note:** This service is available only in Oracle WebCenter Portal: Spaces.	"Working with the Notes Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
Notifications	Provides a means of subscribing to services and application objects and, when those objects change, receiving notification across one or more selected messaging channels.	Chapter 49, "Integrating Notifications"
People Connections	Provides social networking capabilities, such as creating a personal profile, displaying current status, and viewing other users' recent activities.	Chapter 38, "Introducing the People Connections Service"
Polls	Enables you to survey your audience (such as their opinions and their experience level), check whether they can recall important information, and gather feedback.	Chapter 37, "Integrating the Polls Service"
Personalization	Enables you to deliver content within your application to targeted application users based on selected criteria.	Chapter 67, "Personalizing WebCenter Portal Applications"
Recent Activities	Provides a summary view of recent changes to documents, discussions, and announcements.	Chapter 50, "Integrating the Recent Activities Service"
RSS	Provides the ability to access the content of many different web sites from a single location—a news reader.	Chapter 53, "Integrating the RSS Service"
Search	Provides the ability to search services, the application, or an entire site. (This includes integrating Oracle Secure Enterprise Search for WebCenter Portal searches.)	Chapter 45, "Integrating the Search Service"
Tags	Provides the ability to assign one or more personally relevant keywords to a given page or document.	Chapter 44, "Integrating the Tags Service"

Table 2–1 (Cont.) Oracle WebCenter Portal: Services

Service	Description	Chapter
Wiki	Provides the ability for geographically diverse teams to originate and collaborate on web documents.	Chapter 32, "Integrating Wikis and Blogs"
Worklists	Provides a personal, at-a-glance view of business processes that require attention (including a request for document review, and other types of business process that come directly from enterprise applications).	Chapter 51, "Integrating the Worklist Service"

WebCenter provides task flows for these services through the Oracle WebCenter Portal: Services Catalog. For information about how to prepare your application to consume these task flows, see Chapter 7, "Preparing Your Application for WebCenter Portal Services."

How Oracle WebCenter Portal: Services Integrate with Each Other

WebCenter Portal: Services are designed to be integrated with each other. Here are some examples of the ways WebCenter Portal: Services work together:

- The Search service is designed to search any WebCenter Portal service in your application. For example, if you have the Documents service in your application, then the search operation looks for terms in document names and in the documents themselves. The results include tag hits, so any matching tags on your documents and pages are also returned, with relevance based on the quality and frequency of tags.
- The Links service lets you view, access, and associate related information across services. For example, in a discussion forum, you can link to a related discussion forum, document, URL, or event. The Links service also supports bidirectional links between objects, such as a link from a discussion topic to a document and back to the discussion topic.
- The Presence icon can appear in any WebCenter Portal service where a user is indicated. For example, click the author's name on a document to invoke a context menu and send an instant message or mail to that person.
- The Recent Activities service lets you view the most recent additions or changes to the announcements, documents, discussions, and pages in your application.

2.4.1 WebCenter Portal: Services and the REST APIs

You can access many of the WebCenter Portal services through their REST APIs. REST APIs allow you to create rich, interactive client-side applications. REST commands use standard HTTP methods as requests to point to the resource being used. Every request returns a response, indicating the status of the operation. If the request results in an object being retrieved, created, or updated, the response includes a standard representation of that object.

For more information and a complete list of the services that include REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

2.5 Introducing Oracle Application Development Framework (Oracle ADF)

This section briefly introduces Oracle ADF and discusses how WebCenter Portal benefits from Oracle ADF technology.

2.5.1 What Is Oracle ADF?

The Oracle Application Development Framework (Oracle ADF) is an end-to-end application framework that builds on Java Platform, Enterprise Edition (Java EE) standards and open-source technologies to simplify and accelerate implementing service-oriented applications. Used in tandem, Oracle JDeveloper and Oracle ADF give you an environment that covers the full development life cycle from design to deployment, with drag-and-drop data binding, visual UI design, and team development features built in.

2.5.2 Why Is Oracle ADF Important for WebCenter Portal Developers?

You can build portals using a variety of different technologies and approaches, and WebCenter supports a variety of open industry standards for portal-building. Oracle ADF provides a solid foundation for WebCenter Portal development, which allows integration with the entire Oracle Fusion Middleware stack. Oracle WebCenter Portal lets you build portals that plug into this overall architecture, leveraging features provided by the rest of the stack.

2.5.3 What Does Oracle ADF Technology Include?

Oracle ADF uses a set of standard JSF components that include built-in Ajax functionality. Ajax is a combination of asynchronous JavaScript, dynamic HTML (DHTML), and XML. This combination allows requests to be made to the server without fully re-rendering the page. While Ajax allows rich client-like applications to use standard Internet technologies, JSF provides server-side control.

Oracle ADF provides over 150 rich components, including hierarchical data tables, tree menus, in-page dialogs, accordions, dividers, and sortable tables. Oracle ADF also provides data visualization components, which are Flash and SVG-enabled and capable of rendering dynamic charts, graphs, gauges, and other graphics that can provide a real-time view of underlying data. Each component also supports skinning, along with internationalization and accessibility.

ADF task flows represent a critical component for you to understand and use in your application development. They provide a modular approach for defining control flow in an application. Instead of representing an application as a single large JSF page flow, you can break it up into a collection of reusable task flows. In each task flow, you identify application activities, the work units that must be performed in order for the application to be complete. An activity represents a piece of work that can be performed when running the task flow. Task flows also have the advantage of being able to be packaged and deployed with the application, rather than requiring a separate deployment like remote portlets. For more information about task flows, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

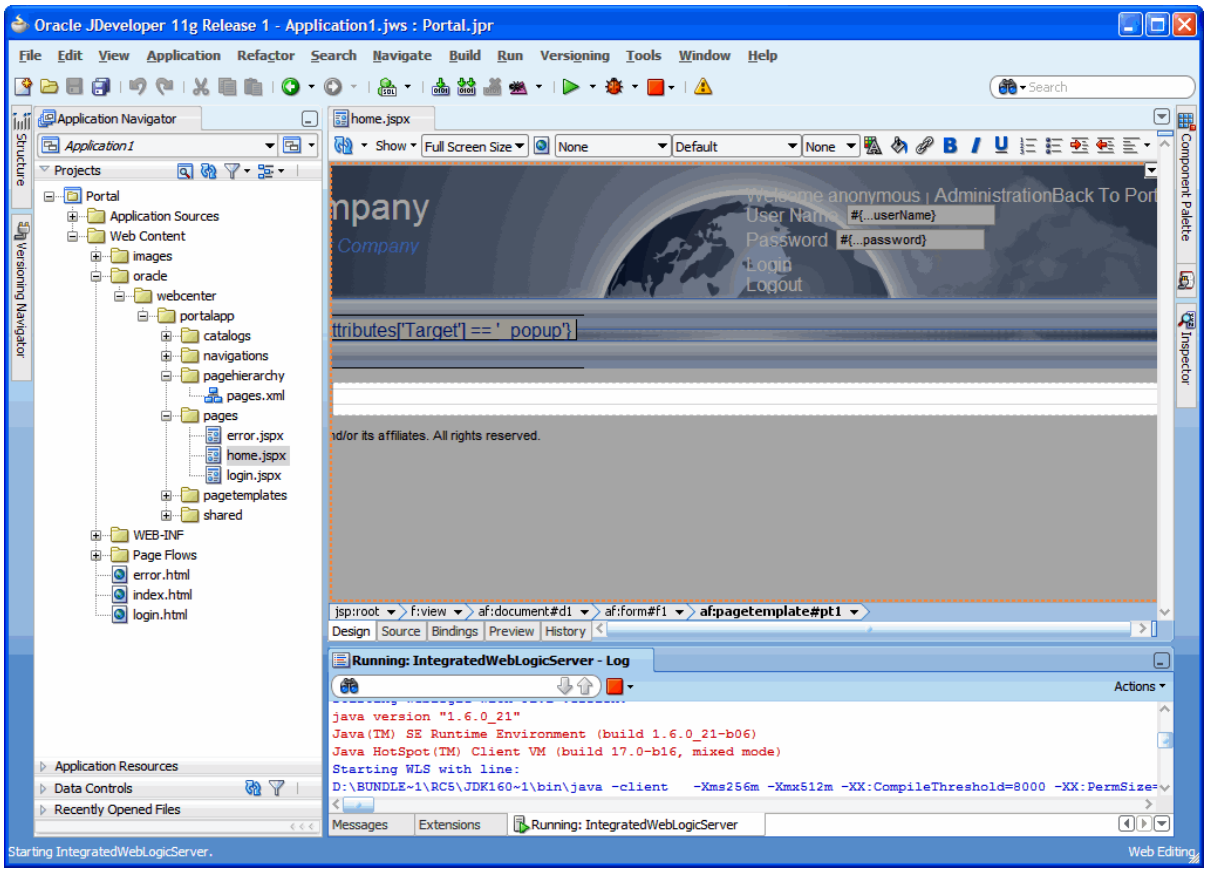
2.6 Introducing Oracle JDeveloper and the WebCenter Portal Extension

Oracle JDeveloper is an integrated development environment (IDE) for building service oriented applications using the latest industry standards for Java, XML, web services, portlets, and SQL. JDeveloper supports the complete software development

life cycle, with integrated features for modeling, coding, debugging, testing, profiling, tuning, and deploying applications. JDeveloper's visual and declarative approach and Oracle ADF work to simplify application development and to reduce mundane coding tasks. For example, code for many standard user interface widgets, such as buttons, lists of values, and navigation bars, are prepackaged for you. All you need to do is select the appropriate widget from the Component Palette and drop it into your application.

WebCenter Portal's extension for Oracle JDeveloper includes all the development functionality you need for building Framework applications. Oracle WebCenter Portal components are readily accessible from a catalog of resources.

Figure 2–2 Oracle JDeveloper and the WebCenter Portal Extension



The WebCenter Portal extension also provides several wizards to help you with essential development tasks such as building portlets, consuming an existing portlet, creating a data control to a content repository, and securing your application. By significantly reducing the amount of coding you do, JDeveloper and the WebCenter Portal extension dramatically increase your productivity as a developer.

In JDeveloper, the easiest way to ensure that you properly define an application and its projects with the appropriate technology scope is to apply an application template. An application template automatically partitions the application into projects that reflect a logical separation of the overall work. The WebCenter Portal extension provides two templates optimally configured for building Framework applications:

- **WebCenter Portal – Framework Application** template. Prepopulates the application with a project and libraries optimally scaled for the creation of data controls and consumption of portlets and Web 2.0 services. Includes Oracle

WebCenter Portal: Framework features like site navigation, page hierarchies, delegated administration, and page templates.

- **WebCenter Portal – Portlet Producer Application template.** Prepopulates the application with a project and libraries scoped for the creation of portlets.

It is not required that you use these templates. If you prefer, you can create your own Framework applications and portlet applications by manually scoping the application technologies and creating the relevant projects. For more information, see Chapter 3, "Preparing Your Development Environment."

For more information about JDeveloper, access the many educational aids from the JDeveloper Start Page, accessible from JDeveloper's Help menu.

2.7 Understanding the Application Life Cycle

The portal life cycle describes the path a portal takes from development through production. Phases of the life cycle include development, testing, staging, and production. For detailed information on the life cycle, see Chapter 10, "Understanding the WebCenter Portal Life Cycle."

2.8 Design Questions to Consider Before You Start

When you design your Framework application, you must consider the needs of your audience. In particular, it is important to think about what features and capabilities your Framework application end users, administrators, and developers need. Before you begin to actually build a Framework application, take a look through the questions listed below and use the answers to help plan your application.

This section includes the following subsections:

- Section 2.8.1, "User Considerations"
- Section 2.8.2, "Site Administrator Considerations"
- Section 2.8.3, "Developer Considerations"

2.8.1 User Considerations

The following list suggests questions you should ask about the end users of your Framework application:

- **How many users are there?** If your Framework application must serve a large number of users, then you must take that into account when you define your deployment environment.That is, you must choose a topology that can support the number of users you expect to access your Framework application and set the configuration parameters of the different services accordingly.

  To help you decide which Oracle Application Server configuration best supports your Framework application, see the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter Portal*.

- **Do users need to edit pages and create enterprise mashups by adding resources?** If you plan to let users edit pages, you must add Oracle WebCenter Portal's Composer components to your application. You should also consider questions such as which areas you want to make editable and what resources to make available to users.

  When you design your application, you must choose where to insert the Composer components to make those areas of the page editable. Components that

allow users to add to the page must appear in a Resource Catalog at runtime. You can also choose to have different catalogs available depending upon the context of the application or the user's privileges.

If you choose to allow page editing, the application must be able to associate the customizations with particular users, which means some form of authentication is required for users of the application. Furthermore, since the user's customizations are stored in Metadata Storage (MDS), you must ensure that your deployment system has access to MDS for the storage and retrieval of customizations.

For more information about the Composer and exposing resource catalogs, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."

- **Do users need to customize their portlets?** Portlets can optionally include a customization mode (Edit mode) that enables an authenticated user to customize the portlet at runtime. The Edit mode might allow the user to enter things such as the portlet's title or a parameter that affects the content of the portlet. For example, you could implement an Edit mode that enables the user to enter ticker symbols for a stock portlet. When the user's changes are applied, the portlet displays the prices for their chosen ticker symbols.

  If you choose to implement customization, then you must also implement some form of security for the application consuming the portlet. Only authenticated and authorized users can customize an application.

  In considering portlet customization, you should also consider whether the portlets are load balanced. If they are load balanced, then you must use database backed customization to ensure that all of your middle tiers can access to the user's customization data. Otherwise, the user has to personalize the same portlet multiple times.

  For more information, see Chapter 58, "Overview of Portlets."

- **Which services do your users need to access?** WebCenter provides a variety of services, such as Documents, Discussions, Search, People Connections, and Instant Messaging and Presence. You must decide which of these services your users need and then configure those services. When the desired services are installed and configured, you can add task flows to your application pages that access those services. The services that you choose to implement can also impact the topology you choose. For example, you might choose to have WebCenter Portal's discussion server and content repositories reside on different systems, which must both be accessible to your application.

  For more information, see Table 2–1, " Oracle WebCenter Portal: Services" and Chapter 7, "Preparing Your Application for WebCenter Portal Services."

- **Will there be both authenticated and unauthenticated (public) users?** Several of the WebCenter Portal services offer limited functionality for unauthenticated users. For example, unauthenticated users can view but not contribute to discussions, and the REST services (most of them) are only available to authenticated users. Due to this loss of functionality, it sometimes works better to create special public pages designed just for unauthenticated users.

  Use the portal page hierarchy editor to set delegated administration security policies on individual pages or on multiple pages. For more information, see Section 9.5.2, "Securing Your Portal Pages."

2.8.2 Site Administrator Considerations

The following list suggests questions you should ask about the administrators of your application:

> **Note:** It is critical that the Framework application administrator and the developers communicate when the application is under construction. At design time, developers must make many choices that determine what the administrator can do to the application at runtime. For example, if the developers choose not to implement skins, then the administrator has no control over the look and feel of the application. Hence, the administrator and the developers should ensure that they consult over these decisions at design time.

- **Do administrators need to customize pages and portlets for users or user groups?** You may want to enable your Framework application administrator to customize pages and portlets to provide default views of the Framework application for users.
  - **Framework application look and feel.** By implementing skins for your Framework application, you enable administrators to choose the look of the pages in the application.

    For information about how to implement skins, see Chapter 19 "Customizing the Appearance Using Styles and Skins" in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework* and Chapter 15, "Creating and Managing Skins."
  - **Customizable Pages.** The administrator can customize the layout of the page.

    For information about customizable components, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."
  - **Portlets.** Portlets can optionally include a customization mode (Edit Defaults mode) that enables the administrator to customize the portlet. The administrator's customization is treated as the default view of the portlet for all other users.

    For more information about portlets, see Chapter 58, "Overview of Portlets."

2.8.3 Developer Considerations

The following list suggests questions you should ask about the developers who will contribute to your application:

- **Do developers and administrators want to use the Oracle WebCenter Portal: Framework features in their application?** Generally, developers should consider using the Framework application template to create framework applications. The template ensures that a new application is provisioned properly with the correct framework technology scopes and features. These portal-specific features include page hierarchies, delegated administration, navigation models, and page templates. In addition, runtime administration tools allow authorized users to modify and change these portal features. For more information Chapter 9, "Understanding Framework Applications."
- **Should developers build task flows or portlets as reusable components?** Thanks to the Oracle JSF Portlet Bridge, developers can make their task flows into JSR 286 portlets. Hence, they have the option of creating reusable components either as

task flows, which can potentially be made into JSR 286 portlets through the bridge, or directly as JSR 286 portlets. If the primary purpose of the component is inclusion in one or more portals, then developers may prefer to build it directly as a JSR 286 portlet, without going through the bridge. If the primary purpose of the component is local execution in a JSF application, then the developer may prefer to create it as a task flow first and then, optionally, use the bridge to run it as a JSR 286 portlet. Building task flows means that the components can be used in other ADF applications without having to repackage them as portlets first.

- **Do developers need to integrate content from content repositories (for example, Content Server, file system, or Oracle Portal)?** If you have content located in content repositories to expose in the Framework application, then you must use JCR adapters and data controls to make that content available.

  For information about including content through JCR data controls and adapters, see Chapter 27, "Managing Content Repository Connections."

- **Will the visible/editable content of the WebCenter Portal: Framework application vary depending upon the identity of the user?** If you plan to have content in your Framework application that is not for everyone, then you must set up a security model with login and user roles that enable you to control access. Components, portlets, and pages can display or not display, depending upon a user's identity. If the content is open to everyone (for example, a Human Resources Framework application with content for all employees), then you might not need to implement access control. The Oracle WebCenter Portal: Framework page hierarchy feature allows you to set security policies on individual or multiple pages in the page hierarchy. For more information, see Section 9.5.2, "Securing Your Portal Pages."

  Furthermore, with Composer, you can allow users to edit the content within the application. Again, you must consider whether you want to restrict that capability to certain users and, if so, you must factor that into your security model.

  For more information about implementing a security model for your Framework application, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."

- **Does your WebCenter Portal: Framework application need to provide access to external applications?** In many cases, you might have external applications to surface in your Framework application. For example, you might have email, Human Resources, or financial applications that you would like users to be able to view from within the application. To provide seamless interaction for the user, you must employ an external application credential store of some kind.

  For more information about accessing external applications from your Framework application, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."

  If you have legacy external application portlets that were built for Oracle Portal, then you can reuse those portlets in a Framework application. This feature allows users to view these portlets from their Framework application rather than having to open each application separately or go back to Oracle Portal. When building such portlets, though, you must remember that they typically must authenticate themselves to the external application before retrieving and displaying any data. As previously mentioned, such authentication requires a credential vault, where the Framework application can store the credentials necessary for logging into the external application. Oracle WebCenter Portal: Framework provides choices in Oracle JDeveloper for incorporating portlets based on external applications.

For more information about external application portlets, see Section 69.13.3, "Managing External Applications."

- **Will the WebCenter Portal: Framework application be developed by a team of developers?** If your Framework application is being developed by a team, then you must think about some additional design considerations. For example, when doing team development, you must be much more aware of which files various JDeveloper actions touch, and may want to take advantage of JDeveloper's ability to integrate code management systems.

  For more information about team development, see Chapter 4, "Working Productively in Teams." See also Chapter 1, "Developer's Quick Start Guide."

2.9 Overview of WebCenter Accessibility Features

Accessibility involves making your application usable by persons with disabilities such as low vision or blindness, deafness, or other physical limitations. In the simplest of terms, this means creating applications that can be used without a mouse (keyboard only), used with a screen reader for blind or low-vision users, and used without reliance on sound, color, or animation and timing.

Oracle software implements the standards of Section 508 and WCAG 1.0 AA using an interpretation of the standards at `http://www.oracle.com/accessibility/standards.html`.

This section describes accessibility features that are specific to WebCenter Portal. For general information about creating accessible ADF Faces pages, see the Developing Accessible ADF Faces Components and Pages section in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*. For information about accessibility features in JDeveloper, see the help topics available by selecting the JDeveloper Accessibility node under JDeveloper Basics in the online help table of contents.

2.9.1 Generating Accessible HTML

WebCenter Portal provides several Composer components that you can add to your application pages to make them editable at runtime. These components provide attributes that are used to generate accessible HTML. To ensure that the pages you create are accessible, you must set these attributes, listed in Table 2–2.

Table 2–2 Accessibility Attributes for WebCenter Portal's Composer Components

Component	Accessibility Attributes
`pe:changeModeButton`	No accessibility attributes.
`pe:changeModeLink`	No accessibility attributes.
`pe:imageLink`	`shortDesc`—Mandatory. This attribute transforms into an HTML `alt` attribute. `accessKey`—Optional. This attribute sets the mnemonic character used to gain quick access to the component.
`pe:pageCustomizable`	No accessibility attributes
`pe:layoutCustomizable`	`shortDesc`—Mandatory. This attribute transforms into an HTML `alt` attribute. `accessKey`—Optional. This attribute sets the mnemonic character used to gain quick access to the component.
`cust:panelCustomizable`	No accessibility attributes

Table 2–2 (Cont.) Accessibility Attributes for WebCenter Portal's Composer

Component	Accessibility Attributes
`cust:showDetailFrame`	`shortDesc`—Mandatory. This attribute transforms into an HTML `alt` attribute.

2.9.2 Accessibility Features at Runtime

When you enable users to customize a page at runtime, you must ensure that any customizations are also accessible to all users. For all components that users can create at runtime, all accessibility-related attributes are shown in the Property Inspector where users can set them appropriately.

For a list of accessibility-related attributes for WebCenter Portal-specific components, see Table 2–2. For a list of accessibility-related attributes for other components, see the Developing Accessible ADF Faces Components and Pages section in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

2.9.3 Accessibility Considerations for Portlets

IFrames are not very well accommodated by today's screen readers and so are not permitted by some accessibility standards.

> **Note:** Portlets created using the Oracle JSF Portlet Bridge have the `requireIFrame` container runtime option set to `true` as these portlets are too complex to render directly on Oracle ADF pages due to JavaScript issues.

WebCenter Portal provides an optional attribute in the `adf:portlet` tag called `renderPortletInIFrame`. You can set this attribute to false to avoid ever using IFrames. For detailed information, see Section 64.4.12, "What You May Need to Know About IFRAMEs."

3

Preparing Your Development Environment

This chapter discusses tasks you may need to perform to set up your development environment.

> **Tip:** If you are just getting started, we recommend that you look at Section 1.1, "WebCenter Portal Developer's FAQ" first.

This chapter includes the following sections:

- Section 3.1, "Installing Oracle JDeveloper"
- Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper"
- Section 3.3, "Setting the User Home Directory Environment Variable"
- Section 3.4, "Working with the Integrated WebLogic Server"
- Section 3.5, "Installing Database Schemas"
- Section 3.6, "Installing the Oracle WebCenter Adapter for SharePoint"

3.1 Installing Oracle JDeveloper

Oracle JDeveloper provides an integrated development environment for developing WebCenter Portal: Framework applications. For information on obtaining and installing Oracle JDeveloper, see the Oracle JDeveloper page on OTN at:

`http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html`

3.2 Installing WebCenter Portal's Extension for Oracle JDeveloper

Before you can begin to develop Framework applications, you need to install the WebCenter Portal's extension for Oracle JDeveloper. The WebCenter Portal extension is a JDeveloper add-in that provides the complete set of WebCenter capabilities and features to the JDeveloper Studio Edition.

To install the WebCenter Portal extension:

1. Start Oracle JDeveloper.
2. If the Select Default Roles dialog displays, select **Default Role** to enable all technologies, and click **OK**.
3. If a dialog opens asking if you want to migrate settings from an earlier version, click **No**.
4. From the **Help** menu, select **Check for Updates**.

5. Click **Next** in the Welcome page of the Check for Updates wizard.
6. On the Source page, under Search Update Centers, search for the **WebCenter Portal extension**, select it, then click **Finish**.
7. When prompted, restart JDeveloper.

JDeveloper is now configured to create WebCenter Portal: Framework and Portlet Producer applications.

For more information on obtaining and installing Oracle WebCenter Portal: Framework, see the Oracle WebCenter Portal page on OTN at:

`http://webcenter.oracle.com`

3.3 Setting the User Home Directory Environment Variable

Oracle strongly recommends that you set an environment variable for the user home directory that is referenced by JDeveloper. By setting this variable, you can avoid receiving long pathname errors that are known to occur in some circumstances.

For detailed instructions on setting this variable on Windows, Linux, UNIX, and Mac OS X operating systems, see "Setting the User Home Directory" in the *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

3.4 Working with the Integrated WebLogic Server

Installation of Oracle WebCenter Portal: Framework reconfigures the Integrated WebLogic Server (WLS) domain in JDeveloper to include additional libraries and several prebuilt portlets. This section discusses the Integrated WLS, including how to start and stop it, and describes some of the preconfigured portlet producers and prebuilt portlets it provides. It contains the following subsections:

- Section 3.4.1, "How to Start and Stop Integrated WLS"
- Section 3.4.2, "What You May Need to Know About Integrated WLS"
- Section 3.4.3, "The WebCenter Preconfigured Server Readme File"
- Section 3.4.4, "What You May Need to Know About Preconfigured Portlet Producers"

3.4.1 How to Start and Stop Integrated WLS

Options for starting Integrated WLS are available in the **Run** menu in Oracle JDeveloper.

- To start Integrated WLS in debug mode, select **Debug Server Instance** from the **Run** menu.

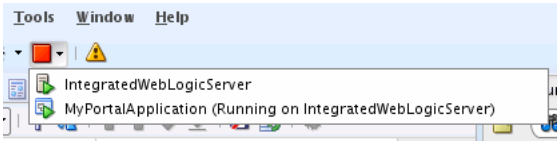
  Running the service in debug mode helps in debugging the service.

- To start Integrated WLS in the regular mode, select **Start Server Instance** from the **Run** menu.

There are a couple of ways to determine if Integrated WLS is running and to stop it.

- The Terminate menu shows you a list of running server(s) and the deployed application(s), if any. (Figure 3–1). To stop a server (or to undeploy an application) select it from this menu.

Figure 3–1 The Terminate Menu Shows What Is Running



- Select **Terminate** from the Run menu, and select the server to stop it.
- Access the Integrated WLS console from a browser:

  `http://localhost:7101/console`

> **Note:** Sometimes WebLogic Server is not accessible (for example, if a user tries to restart WebLogic Server too quickly, before it has successfully shut down). In this case, you may have to manually shut down or stop the Java process.

3.4.2 What You May Need to Know About Integrated WLS

Integrated WebLogic Server (Integrated WLS) is a preconfigured WebLogic Server that provides a complete Java 2 Enterprise Edition (Java EE) 1.4-compliant environment. It is written entirely in Java and executes on the Java Virtual Machine (JVM) of the standard Java Development Kit (JDK). You can run WebLogic Server on the standard JDK provided with your operating system or the one provided with Oracle JDeveloper.

You can use Integrated WLS as a platform for pretesting Framework application deployments on your local computer by establishing an application server connection to it from Oracle JDeveloper. When you run the application in Integrated WLS, it is actually deployed as if you were deploying it to a WebLogic Server instance in an application server. For more information about Integrated WLS, see Section 70.3, "Deploying a Framework Application to a WebLogic Managed Server."

> **Note:** You can specify the Java Virtual Machine (JVM) settings for a Framework application running on the Integrated WLS in the `setDomainEnv.sh` script located here:
>
> *JDEV_SYSTEM_DIRECTORY*`/DefaultDomain/bin/`
>
> The default memory values are:
>
> `-Xmx512m -XX:PermSize=128m -XX:MaxPermSize=512m`
>
> When creating or referring to the `JDEV_SYSTEM_DIRECTORY`, keep in mind that, on a Windows platform, a WebCenter domain name cannot contain spaces, and the domain cannot be created in a folder that has a space in its path. Also, pages in Framework application are not rendered if there is a space in the path to the system directory in Oracle JDeveloper. Therefore, ensure that `DOMAIN_HOME` and `JDEV_SYSTEM_DIRECTORY` paths do not contain spaces.

3.4.3 The WebCenter Preconfigured Server Readme File

The WebCenter Preconfigured Server Read Me file contains valuable information about how to use Integrated WLS. Additionally, it contains links to the preconfigured

portlet producers. You can access the Preconfigured Server Read Me file by selecting **WebCenter Preconfigured Server Readme** from the Oracle JDeveloper Help menu.

3.4.4 What You May Need to Know About Preconfigured Portlet Producers

The WebCenter preconfigured server provides a variety of ready-to-use portlets that you can add to your application pages. Simply register the producers contained in the default server with your Framework application, and then select the producers' portlets from Oracle JDeveloper's Application Resources panel.

This section provides a brief description of the preconfigured producers and some of the portlets they provide. It contains the following subsections:

- Section 3.4.4.1, "OmniPortlet and Web Clipping"
- Section 3.4.4.2, "The WSRP Sample Portlet Producers and Portlets"
- Section 3.4.4.3, "The PDK-Java Sample Portlet Producer and Portlets"

For information about registering portlet producers, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application." For information about adding portlets to pages, see Section 64.3, "Adding Portlets to a Page."

3.4.4.1 OmniPortlet and Web Clipping

The Integrated WLS contains PortalTools, which provides access to the *design time at runtime* OmniPortlet and Web Clipping portlet. Design time at runtime means that users define portlet content after the portlet is placed on an application page and the page is run. This concept is explained more fully in Section 58.2.6, "Portlet Creation Style."

To access OmniPortlet and Web Clipping portlet producers:

1. Start the Integrated WLS (see Section 3.4.1).
2. From the **Help** menu, select **WebCenter Preconfigured Server Readme**.
3. In the Readme file, go to the heading **Preconfigured Portlet Producers**, and click the **PortalTools Welcome Page** link under **PortalTools Portlet Producers**.

   This opens the PortalTools Welcome page.
4. On the PortalTools Welcome page, copy the URL of the Web Clipping Producer link, the OmniPortlet Producer link, or the Sample Portlet Producer link, and use it as the producer URL in the Oracle PDK-Java Portlet Producer Registration Wizard.

> **Note:** For information about registering a portlet producer, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."

Once you have registered a producer, its portlets become available on Oracle JDeveloper's Application Resources panel. In the Application Resources panel, under the Connections node, select a producer name to list its portlets, then drag a portlet onto a Framework application page. (For information about adding portlets to pages, see Section 64.3, "Adding Portlets to a Page.".)

The PortalTools Welcome page contains producer URLs for three producers:

- The **Web Clipping producer** provides the Web Clipping portlet, which is a browser-based declarative tool that enables dynamic reuse of content from another web source. When the source changes, the content in the Web Clipping portlet also

changes. With the Web Clipping portlet, you use a web browser to navigate to the web page that contains the desired content. Using Web Clipping Studio, which is accessed through the portlet, drill down through a visual rendering of the target page to choose the desired content. For detailed information about the Web Clipping portlet, see Chapter 66, "Creating Content-Based Portlets with Web Clipping."

- The **OmniPortlet producer** provides OmniPortlet, which is a declarative portlet-building tool that enables you to build portlets against a variety of data sources, including XML files, character-separated value files (for example, spreadsheets), web services, databases, web pages, and SAP data sources. OmniPortlet users can also choose a prebuilt layout for the data. Prebuilt layouts include tabular, news, bullet, form, chart, or HTML. For information about OmniPortlet, see Chapter 65, "Creating Portlets with OmniPortlet."
- The **Sample Portlet Producer** is useful for illustrating the potential of an OmniPortlet. The sample producer is for demonstration and should not be used to create real-use portlet instances.

3.4.4.2 The WSRP Sample Portlet Producers and Portlets

The Integrated WLS includes sample WSRP portlet producers and portlets you can use with your application. For more information on WSRP portlet producers, see Section 58.2.3.2, "WSRP Producers."

Note: You can find the source code for the sample portlets is in the following EAR file:

JDEV_HOME`/jdeveloper/webcenter/modules/oracle.portlet.server_11.1.1/wsrp-samples.ear`

where *JDEV_HOME* is the location where JDeveloper is installed on your machine.

To access WSRP sample portlet producers:

1. Start Integrated WLS (see Section 3.4.1).
2. From the **Help** menu, select **WebCenter Preconfigured Server Readme**.
3. In the Readme file, go to the heading **Preconfigured Portlet Producers**, and click the link for the **WSRP Tools Portlet Producer** or **Sample Portlet Producer** under **WSRP Portlet Producers**.

   Both links open different WSRP Producer Test Pages—one that has Parameter Form and Parameter Display portlets, the other for different WSRP producer versions of sample portlets.
4. Copy the Web Services Description Language (WSDL) URL for a WSRP producer—either WSRP v1 WSDL or WSRP v2 WSDL.

   Use the copied link as the producer URL in the WSRP Producer Registration Wizard.

Note: For information about registering a portlet producer, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."

Depending on where the portlet producer connection was registered, its portlets appear either under Application Resources or in the Resource Palette. From here, you can drag and drop a variety of sample portlets onto your Framework application pages.

3.4.4.3 The PDK-Java Sample Portlet Producer and Portlets

The Integrated WLS includes sample PDK-Java portlet producers and portlets you can use with your application. Use the PDK-Java sample portlets to familiarize yourself with the types of functionality that are available through PDK-Java portlets.

To access PDK-Java sample portlet producers:

1. Start Integrated WLS (see Section 3.4.1).
2. From the **Help** menu, select **WebCenter Preconfigured Server Readme**.
3. In the Readme file, go to the heading **Preconfigured Portlet Producers**, and copy the link location for the **PDK-Java Sample Producer** or **PDK-Java Struts Sample Producer** link under **PDK-Java Portlet producers**.

   Use the copied link as the producer URL in the Oracle PDK-Java Portlet Producer Registration Wizard.

> **Note:** For information about registering a portlet producer, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application".

Depending on where the portlet producer connection was registered, its portlets appear either under Application Resources or in the Resource Palette. From here, you can drag and drop a variety of sample portlets onto your Framework application pages.

3.5 Installing Database Schemas

To use certain Oracle WebCenter Portal components, you must install schemas into a supported database. This section discusses the schema installation for Oracle, SQL Server, and DB2 databases.

For information on which databases that WebCenter supports, see the document "Oracle Fusion Middleware Supported System Configurations" on OTN.

Table 3–1 lists the schemas that are used with WebCenter and the methods by which you can install them. The Repository Creation Utility (RCU) is the recommended approach for all the schemas. However, for WebCenter Portal's schema, you have the option of installing with a SQL script, as explained below.

Table 3–1 Database Schema Summary

Schema	Description	Installation Method
WebCenter	To use the Tag, Links, Lists, Polls, and People Connections services, you must have WebCenter Portal's schema installed in your database.	▪ RCU (preferred) ▪ SQL script (On Oracle databases only. See below for details.)
Portlets	For storing customizations in portlet producers.	RCU only

Table 3–1 (Cont.) Database Schema Summary

Schema	Description	Installation Method
Activities	For Activity Graph and Analytics products.	RCU only
Discussions/WebCenter Portal Discussions Crawler	Used as the back end for the Discussions and Announcements services.	RCU only

3.5.1 Installing Schemas With RCU

Oracle recommends that you use RCU to install all WebCenter database schemas. For detailed information, see "Creating Schemas" in the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

3.5.2 Installing WebCenter Portal's Schema with a SQL Script

To use the Tag, Links, and People Connections services, you must have the WebCenter Portal schema installed in your database. You can do this by using the built-in SQL Worksheet utility that you used in the previous step.

To install the WebCenter Portal schema:

1. From the **Tools** menu, select **SQL Worksheet**.
2. In the Select Connection dialog, click the pencil icon to edit the connection.
3. Modify the connection to use an administrator username and password, such as `SYS` (using the `SYSDBA` role) then click **OK**.
4. Click **OK** to close the Select Connection dialog.
5. From the Tools menu, choose **SQL Worksheet**.
6. Enter the following SQL statement in the SQL Worksheet panel:

   ```
   @@JDEV_HOME/jdeveloper/jdev/extensions/oracle.webcenter.install/sql/oracle/wc_schema.sql
   ```

   where `JDEV_HOME` is the location where JDeveloper is installed on your machine.
7. Click the **Execute Statement** icon, or press **F9**, to run the script.
8. At the prompt, enter `webcenter` as the name for the schema and a password for the schema, such as `welcome1`. The name of the schema *must* be `webcenter`.
9. If prompted for the Default Tablespace and Temporary Tablespaces, re-enter the default values `users` and `temp`, then accept them.

3.6 Installing the Oracle WebCenter Adapter for SharePoint

Before you configure the Oracle WebCenter Portal adapter for SharePoint as described in Section 27.2.3, "How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint," you must install it in JDeveloper.

The adapter files are located in the **Oracle WebCenter Portal Companion DVD** in the `ofm_wc_generic_jcr_sharepoint_adapter_11.1.1.4.0.zip` file. When you extract this ZIP file to a temporary location, you will find the adapter files in the *TEMP_LOCATION*`/WebCenter/services/content/adapters` directory.

To install the adapter:

1. Start Oracle JDeveloper, if it is not already running.
2. From the **Help** menu, select **Check for Updates**.
3. Click **Next** in the Welcome page of the Check for Updates wizard.
4. On the Source page, select **Install From Local File**, then select `oracle.webcenter.content.jcr.sharepoint.ear` from the *`TEMP_LOCATION`*`/WebCenter/services/content/adapters` directory. This is the temporary directory in which you extracted the contents of the `ofm_wc_generic_jcr_sharepoint_adapter_11.1.1.4.0.zip` file from Oracle WebCenter Portal Companion DVD.
5. Click **Finish**. When prompted, restart JDeveloper.

4

Working Productively in Teams

This chapter contains information for a team development environment, where developers are sharing files in Oracle JDeveloper. It includes information about source (or version) control systems, namely Subversion.

> **Tip:** Typically, one member of the development team creates a new portal application and checks it in to the source code repository. You can create any required connections and check those in as well.

This chapter includes the following sections:

- Section 4.1, "Enabling Source Control on Framework Applications"
- Section 4.2, "Understanding Framework Application Files Affected by Developers"
- Section 4.3, "Implementing Common Requirements One Time"
- Section 4.4, "Portlet Producer Considerations"

4.1 Enabling Source Control on Framework Applications

You can use source control in a Framework application similar to the way you would use source control in any other development environment.

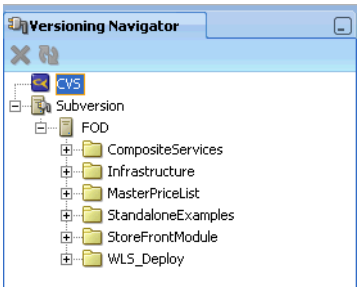
JDeveloper includes Subversion for source control. Oracle also provides free extensions to other source support systems, such as Concurrent Versions System (CVS), Dimensions, and ClearCase. You can install these extensions directly from inside JDeveloper through the **Help - Check for Updates** menu option, which is the recommended way install extensions. If you cannot connect to the internet from your JDeveloper instance, then download the extension from Oracle Technology Network (OTN) at `http://www.oracle.com/technetwork/index.html`. Point the **Check for Updates** wizard to the local file you download.

Note: You may need to configure a proxy server to access the extensions. To set up a proxy server:

1. In Oracle JDeveloper, choose the **Tools** menu option, then select **Preferences**.
2. In the Preferences dialog, scroll down the list and select **Web Browser and Proxy**.
3. In the right pane, under Web Browser and Proxy, select **Use HTTP Proxy Server** and enter the host name and port number of your proxy server, and note any exceptions. Click **OK**.

There are two ways to use Subversion, or any other installed source control system: In JDeveloper, either click the **Versioning** menu or use the Versioning Navigator (click **View - Team - Versioning Navigator**). The Versioning Navigator is shown in Figure 4–1.

Figure 4–1 Versioning Navigator



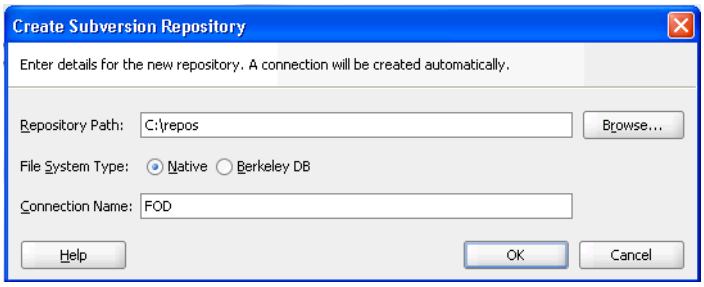
For detailed information about Subversion, see the Oracle JDeveloper online help system and the "Using a Source Control System" section in the *Oracle Application Development Framework Developer's Guide*.

4.1.1 Creating a Subversion Repository

To create a Subversion repository:

1. From the **Versioning** menu, choose Subversion, and then **Create Local Repository**. The Create Subversion Repository dialog displays (Figure 4–2).

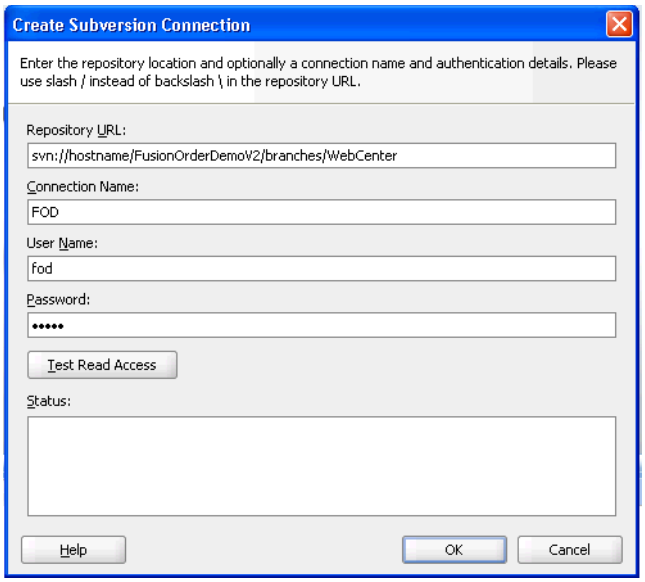
Figure 4–2 Create Subversion Repository



2. In the **Create Subversion Connection** dialog, enter connection details, and then click **OK**. (Figure 4–3)

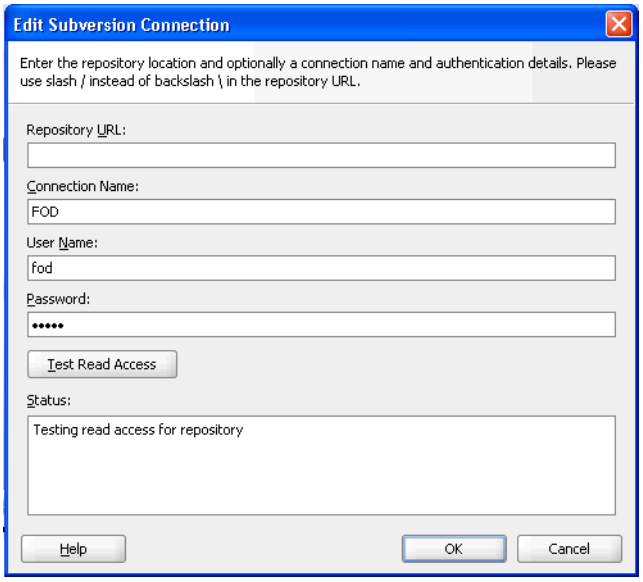
Note: An alternative way to create a connection is to right-click **Subversion** in the Versioning Navigator and select **New Repository Connection**.

Figure 4–3 Create Subversion Connection



3. Later you can edit these pages on the Edit Subversion Connection page (Figure 4–4).

Figure 4–4 Edit Subversion Connection



4.1.2 Understanding Oracle Team Productivity Center

Oracle Team Productivity Center is an Application Lifecycle Management tool that enables software development teams to collaborate and work productively when developing applications using JDeveloper. This also is a free extension to JDeveloper.

Oracle Team Productivity Center integrates task, bug, and defect repositories, such as JIRA, Bugzilla, and its own built-in task repository into the IDE. It provides a chat interface inside JDeveloper and other team working benefits, such as recording details of files checked into the source control system against tasks and developer productivity aids, such as saving the context of your IDE.

For more information, or to download Oracle Team Productivity Center, go to Oracle Technology Network (OTN) at
http://www.oracle.com/technetwork/index.html.

4.2 Understanding Framework Application Files Affected by Developers

An important aspect of working with any source control system is understanding which files are affected by any particular action. Without this knowledge, you may inadvertently corrupt the source by checking in or checking out either too few or too many files given the actions you are performing on the project. This section helps you understand what files are needed for the main actions you may perform while building a WebCenter Portal: Framework application.

This section contains the following subsections:

- Section 4.2.1, "Files Associated With Common Objects"
- Section 4.2.2, "Developer Actions Affecting Metadata Files"

4.2.1 Files Associated With Common Objects

The main objects with which you work in a Framework application are as follows:

- Pages
- Portlets
- Task flows
- Data controls

Each of these are fairly complex objects, made up of several different metadata files.

For information about page metadata files, see the "Introduction to the ADF Metadata Files" section in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*. For information about portlet and producer metadata files, see Appendix A, "Files for WebCenter Portal: Framework Applications."

4.2.2 Developer Actions Affecting Metadata Files

Table 4–1 lists developer actions and the files that are affected by these actions. Take this information into account while managing your files.

Table 4–1 Developer Actions Affecting Metadata Files

Action	Files Affected
Creating or editing the following connection: ■ BPEL ■ Content Repository; for example, Oracle Content Server, Oracle Portal, file system ■ Database ■ Discussions ■ External Application ■ Instant Messaging and Presence ■ Mail ■ Oracle Secure Enterprise Search ■ Oracle PDK-Java Producer ■ WSRP Producer	Adds or updates: ■ *app-root*/.adf/META-INF/connections.xml with the connection ■ *app-root/projectname/projectname*.jpr with connection libraries ■ *app-root*/.adf/META-INF/adf-config.xml with service-level configuration, including the default connection An external application connection, and all connections that require an external application connection (including Mail, Content Repository, and Instant Messaging and Presence) also add or update the following files: ■ *app-root*/src/META-INF/jps-config.xml with the credential store provider ■ *app-root*/src/META-INF/cwallet.sso with the user name and password **Note**: All secure data specified for connections is stored in cwallet.sso, referenced in jps-config.xml. For example, the password for a database connection is stored in cwallet.sso.
Creating a WebCenter Portal data control	Adds or updates following data control files under *app-root/projectname*/adfmsrc: ■ *projectpackage*/DataControls.dcx with the data control definition ■ *projectpackage/dcname*.xml with the data control structure ■ *projectpackage/dcname*/*.xml with data structure definitions used by the data control ■ *app-root/projectname*/adfmsrc/META-INF/adfm.xml with a pointer to the DCX file ■ *app-root/projectname*/adfmsrc/return.xml with the definition for default operations used on collections Updates *app-root/projectname/projectname*.jpr with content repository libraries
Adding or updating pages in a page hierarchy	■ Adds or updates the parent page's page hierarchy file (/oracle/webenter/portalapp/pagehierarchy/*Pages.xml), and updates pages.xml when the parent is the root page of the hierarchy ■ Updates the original parent pages's page hierarchy file when a page is detached from the parent page (for example, when it is removed from the page hierarchy or when it is reparented) ■ Adds or updates the page definition file (*PageDef.xml) for the pages involved (adding a new pagedef file updates *app-root/projectname*/adfmsrc/*projectpackagename*/DataBindings.cpx to add the pagedef and its mapping to the JPSX page)
Adding or updating or removing resources in site resources	Updates mds/oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/generic-site-resources.xml

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Registering an Oracle PDK-Java or WSRP portlet producer	Adds or updates *app-root*/src/META-INF/jps-config.xml with the credential store provider Adds: ■ Tag library ■ *app-root*/mds/producer.pxml (pertaining to the registered producer) ■ A number of files pertaining to the registered producer to *app-root*/mds/directory: producer Adds or updates *app-root/projectname*/adfmsrc/*projectpackagename*/DataBindings.cpx with factory classes to data bindings Updates: ■ *app-root*/.adf/META-INF/connections.xml with a connection to the producer and the URL connection associated with it ■ *app-root*/.adf/META-INF/adf-config.xml with configuration for the metadata repository and the credential store ■ *app-root/projectname*/src/META-INF/orion-application.xml with the WebCenter Portal shared libraries ■ *app-root/projectname/projectname*.jpr with the libraries needed for any Framework application including external application libraries ■ *app-root/projectname*/public_html/WEB-INF/web.xml with the portlet servlets and filters and also external application automated login entries Copies the keystore file to *app-root*/.adf/META-INF if the producer is WSRP and supports WS-Security
De-registering an Oracle PDK-Java or WSRP portlet producer	Updates *app-root*/.adf/META-INF/connections.xml to remove the producer and URL connection Removes producer-related files from *app-root*/mds/ **Note**: Although the producer is removed from the application, you may need to manually remove the metadata files, portlet view tag, and portlet bindings from the application. Everything else should be left alone in case other producers rely on *app-root/projectname*/public_html/WEB-INF/web.xml or *app-root*/.adf/META-INF/adf-config.xml. The metadata files are removed by the deregister operation; however, any customizations on the producer side might need to be manually removed if the producer was not available during deregistration.

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Adding or removing a portlet from a page Adding or removing a portlet on a page from a portlet producer connection in the Resource Palette	When adding a portlet from a portlet producer in the Resource Palette, WebCenter Portal first registers the producer with the application (if it is not registered). See the "De-registering an Oracle PDK-Java or WSRP portlet producer" row for files affected. It then adds or removes the portlet from the page, affecting the files described in this row. Adds or updates: ■ *app-root/projectname*/adfmsrc/*projectpackagename*/DataBindings.cpx with a new PageDef mapping (if it does not exist) ■ *app-root/projectname*/public_html/WEB-INF/faces-config.xml (This happens whenever you add an Oracle ADF component, such as a portlet, to a page.) ■ adf-faces-config.xml (This happens whenever you add an Oracle ADF component to a page.) ■ *app-root/projectname*/public_html/WEB-INF/oracle-webservices.xml if the portlet is WSRP Adds: ■ A number of files pertaining to the newly cloned portlet instance to *app-root*/mds/ ■ A number of WSDL and XML files to *app-root/projectname*/public_html/WEB-INF/wsdl if the portlet is WSRP Updates: ■ *app-root/projectname*/public_html/*pagename*.jspx with the <adfp:portlet> tag (JSF faces tags are replaced with Oracle ADF faces tags.) ■ *app-root/projectname*/adfmsrc/*projectpackagename*/pageDefs/*pagename*PageDef.xml with the region binding and page variable definitions ■ *app-root/projectname*/public_html/WEB-INF/web.xml with secure-ref entries if the portlet is WSRP

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Dragging a file or folder from a content repository connection to a page Dragging a content repository data control on a page	Updates: ■ *app-root/projectname*/public_html/*pagename*.jspx with appropriate ADF faces tags ■ *app-root/projectname*/adfmsrc/*projectpackagename*/pageDefs/*pagename*PageDef.xml with the region binding ■ *app-root/projectname*/adfmsrc/*projectpackagename*/DataBindings.cpx with a listing of the page definitions ■ *app-root/projectname*/adfmsrc/META-INF/adfm.xml with a pointer to the data bindings file ■ *app-root*/.adf/META-INF/adf-config.xml with MDS configuration entries ■ *app-root/projectname/projectname*.jpr with service libraries, dependent libraries, and JSP tag libraries ■ *app-root/projectname*/public_html/WEB-INF/web.xml with ADF library, external application, and MDS configuration entries Adds or updates all connection-related files for any service that requires connections Copies over any required connection that was not present in the application, which updates *app-root*/.adf/META-INF/connections.xml and *app-root/projectname/projectname*.jpr **Note**: The Recent Activities, Search, and Tags services add the search extension to adf-config.xml and create the sample service-definition.xml file (if it does not exist). The service-definition.xml file lets developers define QueryManager implementations for their service.
Creating a JSR 286 portlet application or adding a portlet to an existing application that uses JSP file	Adds or updates: ■ *app-root/projectname*/public_html/WEB-INF/portlet.xml ■ *app-root/projectname*/public_html/WEB-INF/oracle-portlet.xml ■ *app-root/projectname*/public_html/WEB-INF/web.xml Adds Java and JSP source files for the portlet

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Creating a JSR 286 portlet application or adding a portlet to an existing application that uses the ADF/JSPX option	A portlet application generated with this option uses the Oracle Portlet Bridge to access a Faces application. Adds or updates: ■ *app-root/projectname*/public_html/WEB-INF/portlet.xml with configuration information for the portlet bridge ■ *app-root/projectname*/public_html/WEB-INF/oracle-portlet.xml with configuration information for the portlet bridge ■ *app-root/projectname*/public_html/WEB-INF/web.xml with configuration information for a Faces servlet ■ *app-root*/src/META-INF/orion-application.xml with shared library import statements for adf.oracle.domain and oracle.webcenter.skin Adds: ■ Java and JSPX source files for the portlet ■ *app-root/projectname*/public_html/WEB-INF/faces-config.xml
Creating a PDK portlet producer application	Adds or updates: ■ *app-root/projectname*/public_html/WEB-INF/web.xml with producer servlet definitions and producer deployment details ■ service_id.properties with producer configuration attributes ■ *app-root*/src/META-INF/orion-application.xml with shared library import statements for oracle.portal Adds Java and JSP files for the portlet
Registering a secure WSRP producer (WS-Security)	Updates *app-root*/.adf/META-INF/connections.xml with the secure producer connection definition **Notes**: The connections.xml entry for a secure producer looks different in terms of how the security policy is stored. Instead of the whole security policy description, the security policy is just a URI (that is, a "pointer" to the policy in the policy repository). The secure ref attributes in connections.xml are stored in the credential store (as defined by *app-root*/src/META-INF/jps-config.xml). The credential store configuration information (as defined by jps-config.xml) is stored in *app-root*/.adf/META-INF/adf-config.xml. At design time, the credential store is stored in an Oracle Wallet by default; that is, META-INF/cwallet.sso.

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Configuring ADF Security for a project	Adds or updates: ■ *app-root/projectname/*public_html/WEB-INF/web.xml with login configuration, ADF authentication servlet, security constraints, logical roles and mappings ■ *app-root*/src/META-INF/jps-config.xml with identity store, credential store, policy store, anonymous provider, and login modules ■ *app-root/*.adf/META-INF/adf-config.xml with ADF identity store and credential store configuration ■ *app-root*/src/META-INF/jazn-data.xml with an XML identity store ■ *app-root*/src/META-INF/jazn-data.xml with application permissions
Adding a component or service on a page	Updates: ■ app-root/projectname/public_html/WEB-INF/weblogic.xml with data source (WebCenterDS, ActivitiesDS) configuration and WebCenter Portal shared libraries ■ app-root/src/META-INF/weblogic-application.xml with WebCenter Portal shared libraries and XML parser configuration ■ *app-root/projectname/*public_html/WEB-INF/web.xml with data source configuration, context parameters and listeners, JSP servlet, resource servlet, adfBindings filter, and photo servlet entries ■ *app-root/*.adf/META-INF/adf-config.xml with MDS configuration mappings ■ *app-root/projectname/projectname*.jpr with service-specific libraries ■ app-root/projectname/src/META-INF/jazn-data.xml with task flow security grants (if the application has been configured with ADF security) ■ *app-root/projectname/*adfmsrc/*projectpackagename/*pageDefs/*pagename*PageDef.xml with the region binding ■ *app-root/projectname/*adfmsrc/META-INF/adfm.xml with a pointer to the data bindings file ■ *app-root/projectname/*adfmsrc/*projectpackagename/*DataBindings.cpx with the task flow binding definition factory and pageDef entries ■ *app-root/projectname/*public_html/*pagename*.jspx with appropriate ADF faces tags **Notes**: Some services also affect additional files or have a design time dependency on other services. These services are listed in the following rows. The *projectname*.jpr and web.xml files are updated with service libraries and dependent libraries.
Activity Streaming	Dependent Services: External Application, IMP

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Announcements	Updates: ■ `weblogic-application.xml` with the discussions shared library ■ *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings and service-specific configuration entries Dependent Services: External Application, IMP, Links, Search
Composer	Updates: ■ *`app-root/projectname/`*`public_html/WEB-INF/web.xml` with context parameters ■ *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings and ADF faces, resource bundle, and catalog configuration entries Adds `$PROJECT_DIR/src/<projectpackagename>/default-catalog.xml`
Discussions	Updates: ■ `app-root/src/META-INF/weblogic-application.xml` with the discussions shared library ■ *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings and service-specific configuration entry Dependent Services: External Application, IMP, Links, Search
Documents, Wikis and Blogs	Updates: ■ *`app-root/projectname/`*`public_html/WEB-INF/web.xml` with ADF library filter/servlet, GetHandler servlet, and filter mappings ■ *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings Dependent Services: External Application, IMP, Links, Search, Tags
External Application	Updates *`app-root/projectname/`*`public_html/WEB-INF/web.xml` with the ExtAppLogin servlet and filter mappings
Events	Updates *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings and service-specific configuration entry Dependent Services: External Application, Links, Search
Instant Messaging and Presence (IMP)	Updates: ■ `weblogic-application.xml` with the SDP shared library ■ *`app-root/`*`.adf/META-INF/adf-config.xml` with service-specific configuration entry Dependent Services: External Application
Links	Updates *`app-root/`*`.adf/META-INF/adf-config.xml` with service-specific configuration entry
Lists	Updates *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings Dependent Services: External Application, IMP, Links, Search
Mail	Updates *`app-root/`*`.adf/META-INF/adf-config.xml` with MDS mappings and service-specific configuration entry Dependent Services: External Application, IMP

Table 4–1 (Cont.) Developer Actions Affecting Metadata Files

Action	Files Affected
Page	Updates: ■ *app-root/projectname*/adfmsrc/*projectpackagename*/DataBindings.cpx with attributes ■ *app-root*/.adf/META-INF/adf-config.xml with MDS mappings Dependent Services: Links, Search, Composer
People Connections	Dependent Services: External Application, IMP, Composer, Activity Streaming
Polls	Updates *app-root*/.adf/META-INF/adf-config.xml with MDS mappings and service-specific configuration entry Dependent Services: External Application, IMP
Recent Activity	Updates *app-root*/.adf/META-INF/adf-config.xml with service-specific configuration entry Dependent Service: Search
RSS	Updates ■ app-root/src/META-INF/weblogic-application.xml with the discussions shared library ■ *app-root*/.adf/META-INF/adf-config.xml with MDS mappings Dependent Service: External Application
Search	Updates *app-root*/.adf/META-INF/adf-config.xml with MDS mappings and service-specific configuration entry Adds $WORKSPACE_DIR/.adf/META-INF/service-definition.xml
Tags	Updates *app-root*/.adf/META-INF/adf-config.xml with service-specific configuration entry Dependent Services: External Application, IMP, Search, Links
Worklist	Updates app-root/src/META-INF/weblogic-application.xml with the SOA shared library Dependent Services: Search, Links

4.3 Implementing Common Requirements One Time

It is good practice for the project administrator to implement any common developer requirements one time and then check in that version for all to use. By planning ahead and having the administrator take care of these common requirements up front, you can reduce redundancy and error.

For example, suppose two developers must add OmniPortlet on different pages of an application. If the project administrator has registered the OmniPortlet producer, then it is available for both of them to use. If not, then each developer likely registers the OmniPortlet producer separately, leading to unnecessary duplication and confusion.

Another example: suppose many developers need content from the same content repository. One person should set up and check in the needed connection first. Other developers then can simply reuse the same data control.

4.4 Portlet Producer Considerations

When working in a team environment, bear in mind the following considerations pertaining to portlet producers:

- Section 4.4.1, "Portlet Producer Connections"
- Section 4.4.2, "Portlet Producer Name Clashes"
- Section 4.4.3, "Combining Portlets from Different Portlet Producers"

4.4.1 Portlet Producer Connections

When building an EAR file for your project, connections are loaded for the entire application rather than individual projects. For example, suppose you have an application with two projects: P1 and P2. P1 has 100 registered producers and P2 has no producers. When you build an EAR file for either project, all 100 of P1's producer connections are loaded into `connections.xml`. Note, though, that you can also edit `connections.xml` manually.

When you run the predeployment tool to create a targeted EAR file, all of the connections in `connections.xml` must be accessible. Hence, in our example, the generation of a targeted EAR file for either project would fail if any of the 100 portlet producer connections for P1 are unavailable for some reason. Given this behavior, you must carefully consider how you plan to subdivide your overall development effort into applications and projects.

4.4.2 Portlet Producer Name Clashes

While Oracle WebCenter Framework lets you register two portlet producers under the same name, it generally is better to avoid this situation. For example, if two developers working on the same application inadvertently register a portlet producer with the same name, then usually it is best to change one name to be unique. If you have two portlet producers registered under the same name, then it becomes very difficult to distinguish between them when debugging errors or performing administrative tasks on the portlet producers.

4.4.3 Combining Portlets from Different Portlet Producers

In some cases, you might have multiple developers building portlets and ultimately you want those portlets to be combined under a single portlet producer. The developers must be conscious of some potential issues.

- Portlet names and JSP paths could clash. Portlet developers should use prearranged class package names and JSP paths to avoid naming clashes when portlets are combined within one portlet producer in one application.
- When you create a JPS portlet in the Portlet wizard, the directory for the portlet modes defaults to `portlet`*`n`*`\html\`*`mode_name`*, where *n* is a number that increments for each portlet you create. To avoid directory and file name clashes, portlet developers should change the directory name on the Content Type and Portlet Modes page of the Portlet wizard. Select the portlet mode and then change the directory name in the corresponding field to something unique.
- When you combine portlets into one portlet producer, you must manually merge the portlet descriptor files, avoiding identifier clashes as you do so as follows:
  - JPS portlet identifiers in the `portlet.xml` and `oracle-portlet.xml` files are generated automatically starting from `portlet1`. When you manually

merge multiple portlet descriptor files into one, you must change any portlet identifiers that clash with one another.

  - Similarly, the PDK-Java portlet identifiers in `provider.xml` are automatically generated starting from `1`. When you manually merge multiple `provider.xml` files, you must change any portlet identifiers that clash with one another.

- You must manually merge any web descriptor (`web.xml`) changes; for example, security role information.
- For PDK-Java portlets, you also might need to manually merge `.properties` files.
- When using many different portlets, parts of files might not be checked in properly because they are overwritten by someone else or because JDeveloper does not pick up the changed files correctly. For example, you might deploy an application with many types of portlets and see the following error in two of the three OmniPortlets:

  ```
  mdsId
  oracle/adf/portlet/OmniPortlet_1172537210873/ap/Portlet100_0b4156e9_0111_1000_
  8001_82235f273a39.pxml not found

  mdsId
  oracle/adf/portlet/OmniPortlet_1172537210873/ap/Portlet100_250498fc_0111_1000_
  8002_a9fe7286a54e.pxml not found
  ```

  If you receive error messages like these, then ensure that all `*.pxml` files from the development environment are copied over to the deployed environment.
- Your portlet customizations might not appear in your deployed environment. Make sure that the content of the `oracle\adf\portlet\<`*`portletInstanceName`*`>.pxml` file is correct and that it refers to the proper *`*.pxml`* files.

5

Creating Framework and Portlet Producer Applications

WebCenter Portal provides templates for creating two kinds of applications: Framework applications and Portlet Producer applications. A Framework application is an ADF web application that is configured for building portal Web applications that include navigations, social computing services, and runtime customization. A Portlet Producer application is configured for building and deploying standards-based portlets.

This chapter includes these topics:

- Section 5.1, "What Are Application Templates?"
- Section 5.2, "Creating a Framework Application"
- Section 5.3, "Adding Pages to a Portal"
- Section 5.4, "Understanding the PortalWebAssets Project"
- Section 5.5, "Creating a Portlet Producer Application"
- Section 5.6, "How Is a Framework application Security Configured by Default?"

5.1 What Are Application Templates?

To readily find and use the appropriate components in your Framework or Portlet Producer application, you must ensure that the right technology scopes are set, tag libraries added, and required Java classes are added to the class path. Once you do this, relevant components are included in the Component Palette and relevant context menus become available in JDeveloper.

Fortunately, Oracle provides out-of-the-box application templates to ensure that scopes are set properly and the right tag and Java libraries are added to create Framework or Portlet Producer applications.

Using the **WebCenter Portal – Framework Application** template generates default WebCenter Portal: Framework projects with unique technology scopes, libraries, and default files. Additionally, the template limits the types of options that are exposed in the JDeveloper user interface to those appropriate for Framework projects.

> **Note:** In addition to populating the application with default files and folders, the template populates the Resource Palette with the WebCenter Portal – Framework Catalog and the WebCenter Portal – Services Catalog. The Services Catalog contains data controls and task flows that you can use to integrate WebCenter Portal services in your application. The WebCenter Portal – Framework Catalog contains a collection of components that are most commonly used for building portals. For more information see Chapter 16, "Creating and Managing Resource Catalogs."

Application templates provide a means of easily creating an application with a pre-defined set of projects and technology scopes. Templates also provide a means of partitioning an application into projects that reflect a logical separation of the overall work.

The application templates include:

- **WebCenter Portal – Framework Application**: Populates the application with a *portal project* and a *static application resources* project. The portal project includes features like site navigation, page hierarchies, delegated administration, security, page templates, and runtime customization. A Framework application can consume portlets, incorporate content management services, and include WebCenter social computing services. The static application resources project includes static application resources like HTML and image files. By default, this project is called PortalWebAssets. By separating the static resources into a separate project, it is possible to deploy those resources to a dedicated server. See Section 5.2, "Creating a Framework Application" and Section 5.4, "Understanding the PortalWebAssets Project."
- **WebCenter Portal – Portlet Producer Application**: Populates the application with a project scoped for creation of JSR 286 (standards-based) and Oracle PDK-Java portlets. See Section 60.1.2, "Introduction to PDK-Java Portlets."

It is not required that you use these templates. If you prefer, you can create your own Framework or Portlet Producer applications by manually scoping the application technologies and creating the relevant projects. For more information, see Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."

If you are working with an existing application that was created with some other template, you can also extend it with WebCenter Portal components. To do this, you must manually add any required projects, technology scopes, and tag libraries. For information about extending existing applications, see Section 6.2, "Extending Non-WebCenter Portal: Framework Applications to Include WebCenter Capabilities."

This section describes how to create an application using the WebCenter Portal – Framework Application template.

> **Note:** Technology scopes control the options that appear by default in the New Gallery dialog. They can also be used to control the context menus that appear in JDeveloper and to add tag libraries to the project so that tags appear in the Component Palette.

5.2 Creating a Framework Application

This section explains how to create a WebCenter Portal: Framework application. A Framework application is a portal created with a special template called the

WebCenter Portal – Framework Application template. For more information about portals, see Chapter 9, "Understanding Framework Applications."

- Section 5.2.1, "How to Create a Framework Application"

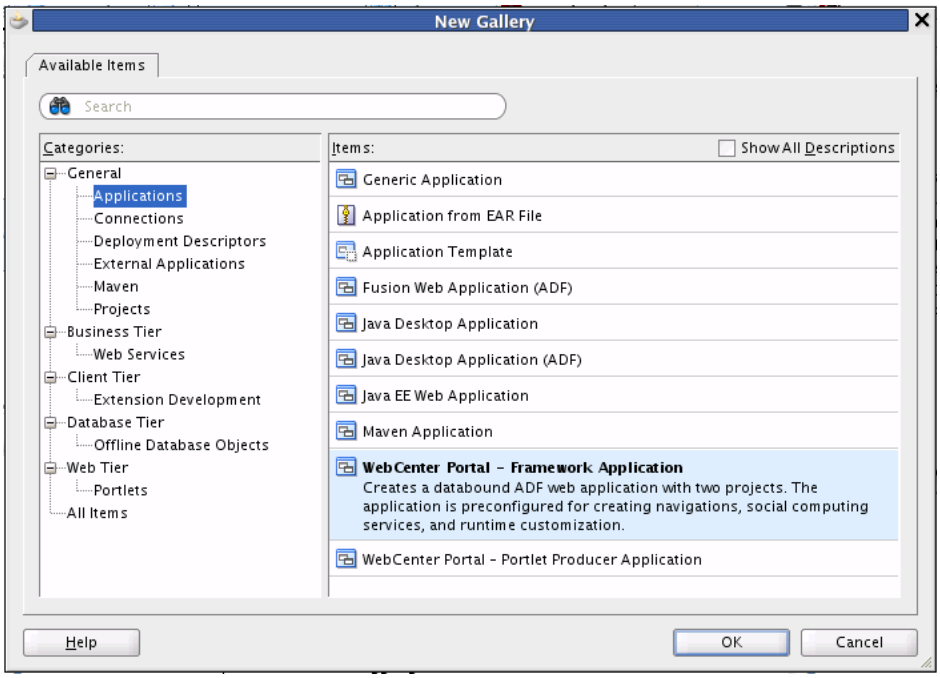
> **Tip:** The **WebCenter Portal – Framework Application** template prepopulates the Framework application with a single portal project that includes features like site navigation, page hierarchies, delegated administration, security, page templates, and runtime customization. For more information, see Section 9.12, "How are WebCenter Portal Files Organized?."

5.2.1 How to Create a Framework Application

A Framework application is a portal you create using the WebCenter Portal – Framework Application template. To create a portal with this template:

1. Access the application creation wizard. JDeveloper provides several ways for you to access this wizard. For example:
   - From the **File** menu, choose **New**. In the New Gallery dialog, expand **Applications**, select **WebCenter Portal – Framework Application**, and click **OK**. For example, see Figure 5–1.

Figure 5–1 New Gallery Dialog



   - From the Application menu, select **New**. In the dialog select **WebCenter Portal – Framework Application**.
   - If you have an existing application that is open, in the Application Navigator, right-click **<*Application-Name*>** and choose **New**. In the New Gallery dialog, expand **Applications**, select **WebCenter Portal – Framework Application**, and click **OK**.
   - If you have an existing application that is open, in the Application Navigator, from the **<*Application-Name*>** drop down menu, select **New**. In the New

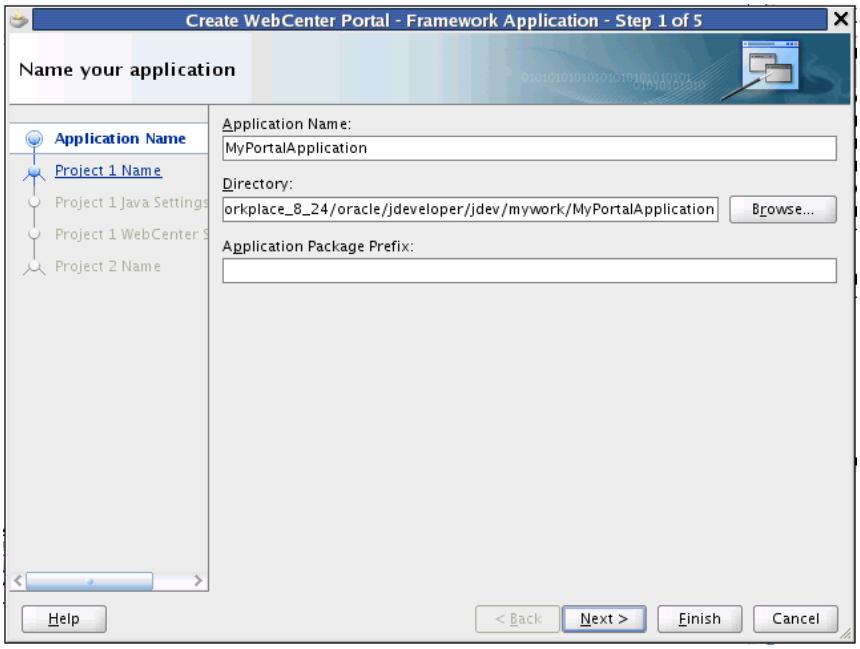
Gallery dialog, expand **Applications**, select **WebCenter Portal – Framework Application**, and click **OK**.

Note: At this point, you are on the first page of the Create WebCenter Portal – Framework Application wizard. The wizard includes four pages, which are explained in the following steps.

---

2. Enter a name for the application in the **Application Name** field, as shown in Figure 5–2.

Figure 5–2 Create WebCenter Portal – Framework Application Wizard Step 1 of 5



3. In the **Directory** field, enter a path to the directory where you wish to store the application, or accept the default path.

   For example:

   ```
   C:\JDeveloper\mywork\Application1
   ```

   Optionally, click the **Browse** button to navigate to the desired directory.

4. If required, in the **Application Package Prefix** field, enter a prefix to use for packages created within this application.

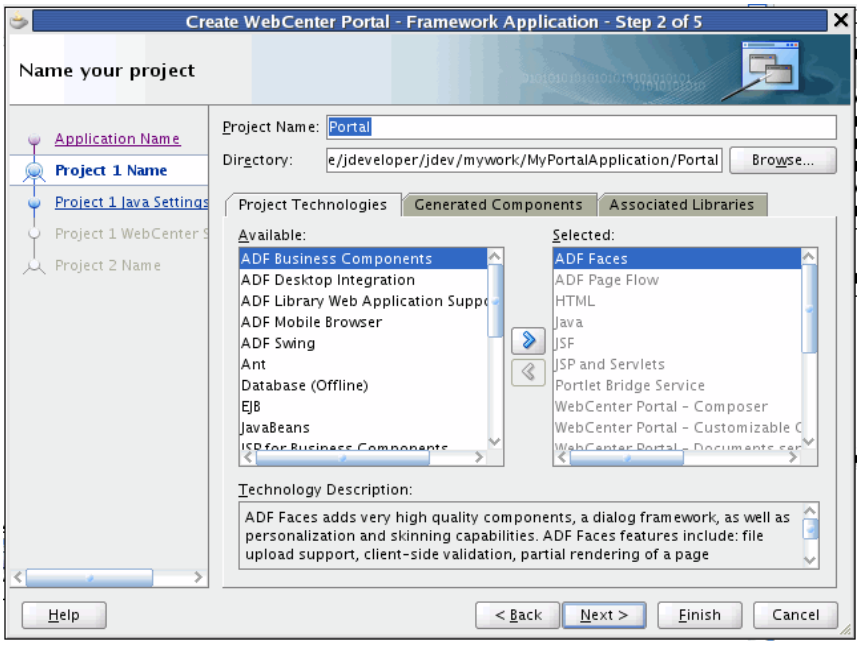
Note: Depending on how you launched the wizard, you might see an Application Template list. If you do see this list, make sure **WebCenter Portal – Framework Application** is selected.

Tip: At this point, you could click **Finish** to create a project with default Framework features, including page hierarchies and navigations. We'll continue and review the rest of the wizard pages.

5. Click **Next**.

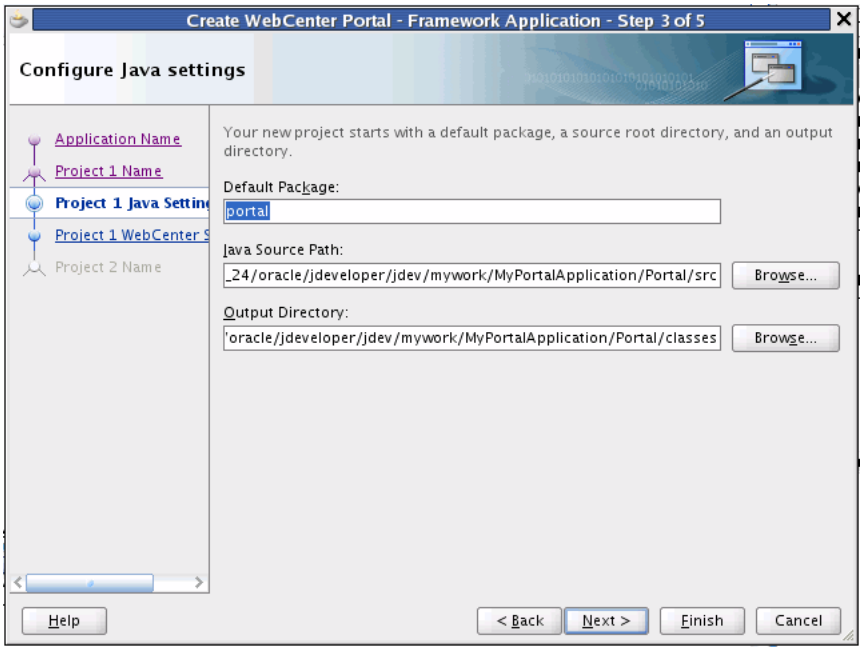
6. Enter a project name, as shown in Figure 5–3.

Figure 5–3 Create WebCenter Portal – Framework Application Wizard Step 2 of 5



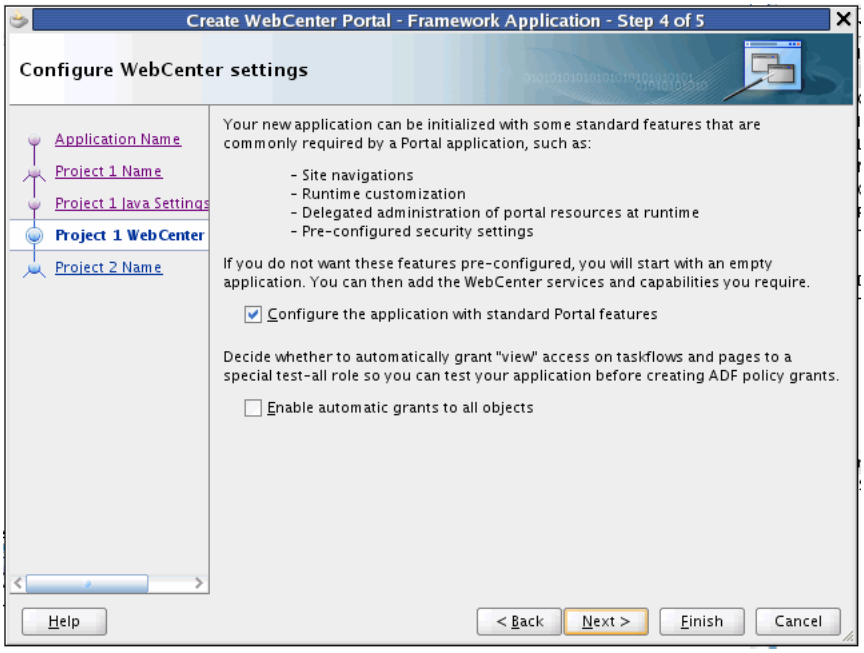
7. Enter a directory in which to place the project, or use the **Browse** button to select a directory.
8. Optionally add or remove **Project Technologies**. For more information, see Section 6.1.3, "What Are the Default Technology Scopes for a Framework Application?."
9. If you wish, select the **Generated Components** and **Associated Libraries** tabs to view additional configuration files and libraries that will be added to the project. For more information, see Section 1.10, "What Configuration Files Should I Know About?."
10. Click **Next**.
11. If you wish, enter a Default Package name and directories for the default Java source files and output class files, as shown in Figure 5–4.

Figure 5–4 Create WebCenter Portal – Framework Application Wizard Step 3 of 5



12. In the next wizard page, two check boxes appear, as shown in Figure 5–5. Generally, the default settings are recommended; however, see the following note for more information on this wizard page.

Figure 5–5 Create WebCenter Portal – Portal Framework Wizard Step 4 of 5



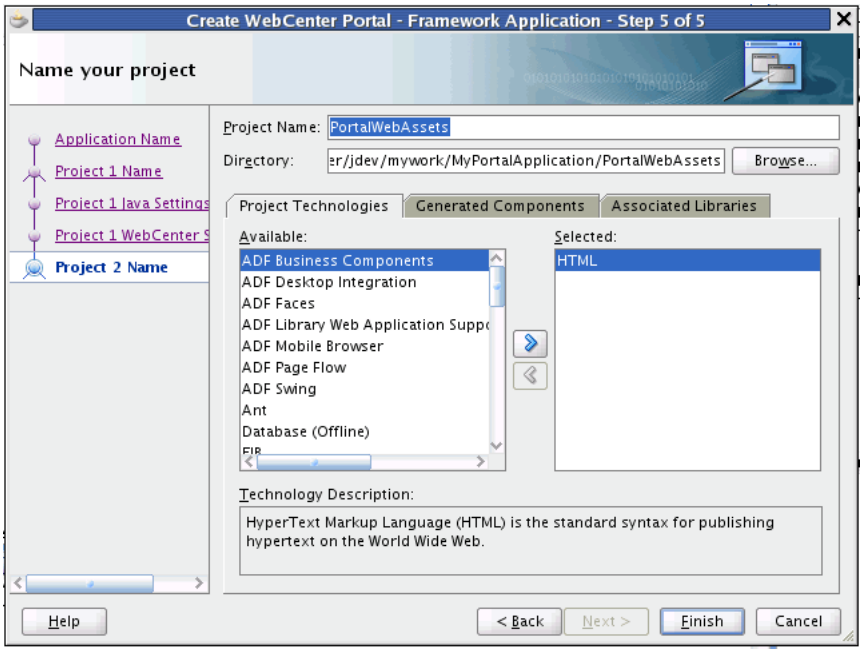
Note: If you want your application to include the features of Oracle WebCenter Portal: Framework, like page hierarchies and navigations, leave the first box checked (**Configure the application with standard Portal features**). If you uncheck this box, the result is that you will create a traditional Oracle Fusion Web Application that does not contain Framework features. For detailed information on the effect of checking or unchecking this box, see Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."

Oracle recommends that you leave the first checkbox selected, unless you know for sure that you do not want to use the Oracle WebCenter Portal: Framework features.

The second checkbox, **Enable automatic grants to all objects**, allows you to create a special role called test-all with View access on all taskflows and pages. This option is primarily added to make testing the application easier. It is unchecked by default.

13. Click **Next**.
14. In the final wizard page you are given a chance to configure the static application resources project. By default, this project is called PortalWebAssets. The project includes static application resources like HTML and image files. By separating the static resources into a separate project, it is possible to deploy those resources to a dedicated server. See For more information, see Section 5.4, "Understanding the PortalWebAssets Project.".

Figure 5–6 Create WebCenter Portal: Framework Application Wizard Step 5 of 5



15. Click **Finish** to create the Framework application.

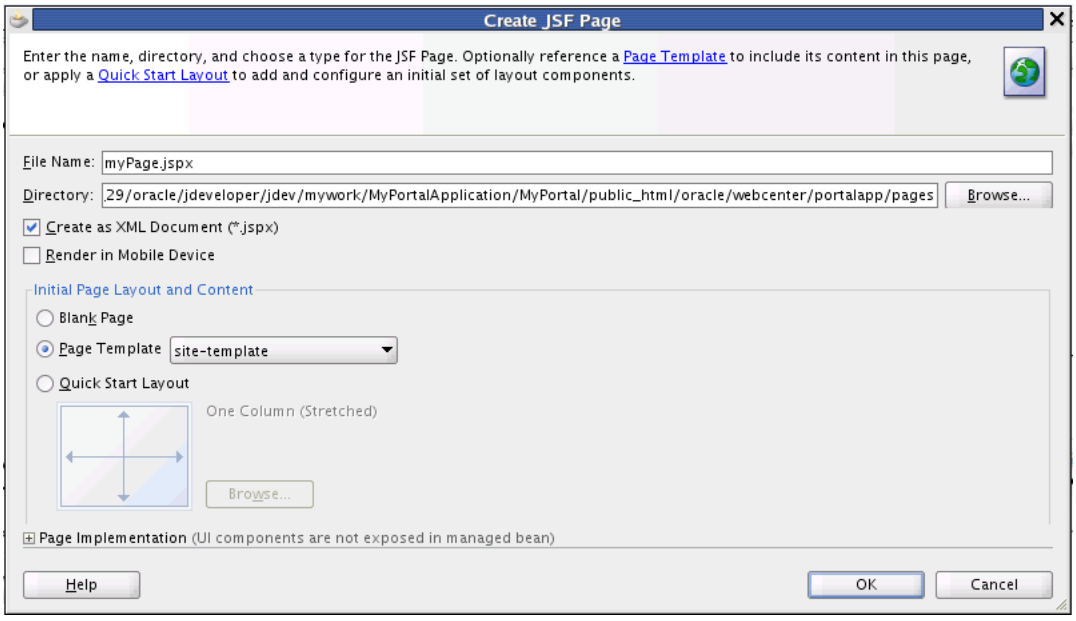
5.3 Adding Pages to a Portal

The WebCenter Portal – Framework Application template populates a new portal application with several pages. This section explains how to add new pages to your portal application. See also Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

> **Note:** Always create portal pages in JDeveloper as JSP *documents* (`.jspx`) rather than JSP *pages* (`.jsp`). For page customizations to be stored, the page must be represented in XML. By choosing to create an XML representation of the JSP document (`.jspx`), you ensure that customizations are always possible for the Framework application page.

1. Start JDeveloper.
2. In the Application Navigator, right-click the `pages` folder (`Web Content/oracle/webcenter/portalapp/pages`) in your portal project and select **New**.
3. In the New Gallery, expand **Web Tier,** select **JSF** and then **JSF Page,** and click **OK.**
4. In the Create JSF Page dialog (Figure 5–7), enter a file name for the page.

Figure 5–7 The Create JSF Page Dialog



5. Select **Create as XML Document (*.jspx).**
6. Select a page template. For more information on page templates, see Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."
7. Click **OK**.

For information about the other settings in this dialog that are not specific to Framework application pages, see the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

5.4 Understanding the PortalWebAssets Project

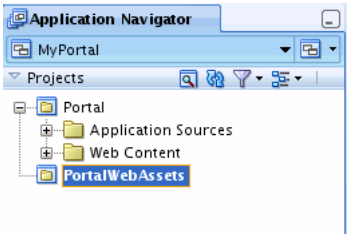
This section discusses the PortalWebAssets project. If you wish, you can place static resources like HTML files and images in this project. The PortalWebAssets project is created, by default, in new Framework applications.

- Section 5.4.1, "Introduction"
- Section 5.4.2, "Decoupling the Static Application Resources Project"
- Section 5.4.3, "Deployment Options"
- Section 5.4.4, "Using EL to Manage Static Resources"

5.4.1 Introduction

Static resources for a web application, like HTML and image files, are typically bundled and deployed with the application. Requests for both dynamic and static resources are generally handled by the host application server. One way to optimize server performance is to separate the static resources from the application and deploy them on a different (possibly less expensive and more scalable) server. Framework applications support this scenario with a static application resources project, which can include static resources like HTML and images. By default, this project is called PortalWebAssets, but you can rename the project if you wish. See Figure 5–8.

Figure 5–8 The PortalWebAssets Folder



To use the static resources project, simply add content (like HTML or image files) to the project. You can do this directly from the file system, by adding files under the `../PortalWebAssets/public_html` directory, or by adding in JDeveloper. For instance, to add an HTML file, select **New** from the File menu, and use the New Gallery wizard to create the resource.

> **Note:** CSS files used for skins must remain in the Framework application project. Do not attempt to place skin CSS files in the static application resources project.

5.4.2 Decoupling the Static Application Resources Project

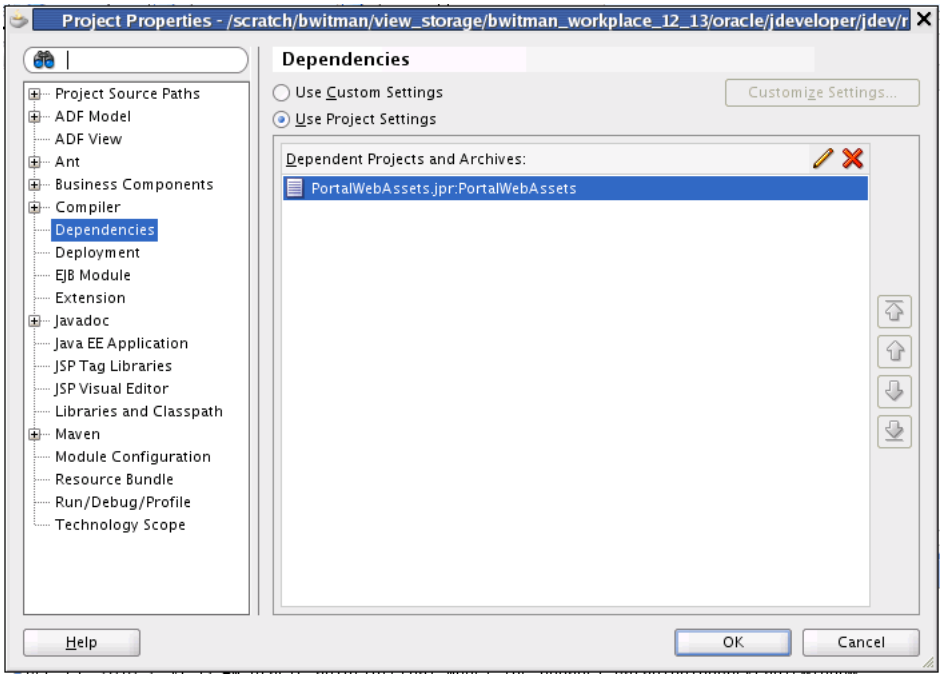
By default, a portal project includes the static application resources project (PortalWebAssets) as a dependency. This means they are built together and deployed to the same server. If you do not wish to deploy static resources to another server, then you do not need to take any action with respect to the static resources project.

If you wish to decouple the PortalWebAssets project from the Portal project, do the following:

1. Right-click the Portal project and select **Project Properties**.

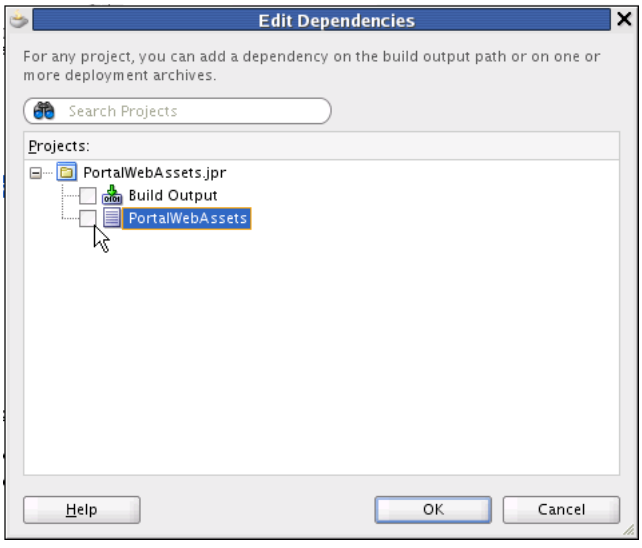
2. In the Project Properties dialog, select the **Dependencies** node, as shown in Figure 5–9.

Figure 5–9 Project Dependencies Dialog



3. To remove the dependency, click the **Delete** button, or select the **Edit** button and deselect the project in the Edit Dependencies dialog, as shown in Figure 5–10.

Figure 5–10 The Edit Dependencies Dialog



4. Generate two distinct archives (`Portal.ear` and `PortalWebAssets.jar`), and deploy them to separate servers. See Section 5.4.3, "Deployment Options."

5.4.3 Deployment Options

The `Portal.ear` file is typically deployed to the application's managed server. The `PortalWebAssets.jar` file can be deployed to any J2EE compliant server container, usually by creating a WAR deployment profile in the PortalWebAssets project and including it in an EAR deployment profile. Another, simpler, option is to unzip the JAR file into an Apache web server. See also Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application."

5.4.4 Using EL to Manage Static Resources

Developers can use EL expressions to dynamically generate the target URL for static resources. You have two options for using EL. With the first option, you need to define a base URL preference. The second allows you to map URLs dynamically.

5.4.4.1 Defining a Base URL Preference

This solution provides a single preference that uses a "base" URL to redirect resources to a desired server. With this option, EL expressions take a format that is illustrated by the following sample:

```
<af:image source="#{preferenceBean.baseResourceURL}/images/globe.png"/>
```

To configure this option, add a preference to the `adf-config.xml` file as follows:

```
<portal:preferences>
...
    <portal:preference id="oracle.webcenter.portalapp.baseresourceurl"
        desc="Default Base Resource URL EL"
        value=
       "#{request.scheme}://#{request.serverName}:#{request.serverPort}#{request.contextPath}"
       resourceType="BaseResourceURL" display="true"
    />
...
</portal:preferences>
```

The only limitation of this technique is that you can only map values to one server. There is only one preference value that can be returned. If you want to dynamically configure URLs, choose the technique described in the next section, Section 5.4.4.2, "Mapping URLs Dynamically."

5.4.4.2 Mapping URLs Dynamically

This solution lets you construct URLs dynamically through the ability to specify multiple namespaces. With this option, resources become a parameter to the preference, allowing the resources to be mapped to the correct namespace.

This solution supports two ADF component categories:

The first category includes static resources referenced by ADF Faces components. This type of call is illustrated by the following sample:

```
<af:image
source="#{preferenceBean.staticResourceURL['/oracle/webcenter/portalapp/static/ima
ges/globe.png']}"/>
```

The second category includes static resources referenced through the `inlineStyle` of ADF Faces components. This type of call is illustrated by the following sample:

```
<af:panelBorderLayout id="pt_pgl1"
inlineStyle='background-image:url(#{preferenceBean.staticResourceURL["/oracle/webc
```

```
enter/portalapp/static/images/globe.png"]});'>
```

The EL expression:

```
#{preferenceBean.staticResourceURL['<resource path>']}
```

performs a resource namespace lookup to derive the resource URL prefix. The input to the lookup is the path of the resource (for example: `/oracle/webcenter/portalapp/static/images/globe.png`). To establish the mappings between resource namespaces and URLs, edit the `<portal:resource-mappings>` section of the `adf-config.xml` file.

To configure this option, add a preference to the `adf-config.xml` file as follows:

```
<portal:adf-portal-config>
    <portal:preferences>
    ...
    </portal:preferences>
    <portal:resource-mappings>
        <portal:resource-mapping path="/"
 url-prefix="#{request.scheme}://#{request.serverName}:#{request.serverPort}#{request.contextPath}"
       />
    </portal:resource-mappings>
</portal:adf-portal-config>
```

The lookup yields the prefix of the resource URL. The final value of the resource URL is derived by concatenating the URL prefix with the path to the resource. For example,

```
/oracle/webcenter/portalapp/static/images/globe.png
```

yields:

```
http://myserver/static_resources/oracle/webcenter/portalapp/static/images/globe.png
```

At development time, the url-prefix lookup always resolves to the application's context URL, for example: `http://<server>:<port>/<context root>`.

Example 5–1 shows a sample mapping configuration in `adf-config.xml`:

Example 5–1 Sample Namespace Lookup Mapping

```
<portal:adf-portal-config>
    <portal:preferences>
    ...
    </portal:preferences>
    <portal:resource-mappings>
        <portal:resource-mapping path="/oracle/webcenter/portalapp/static"
            url-prefix="http://myserver/static_resources"/>
        <portal:resource-mapping path="/mycompany/static/images"
            url-prefix="http://myserver/static_resources"/>
        <portal:resource-mapping path="/mycompany/myapp/static/images"
            url-prefix="http://myserver/static_resources/myapp2"/>
    </portal:resource-mappings>
</portal:adf-portal-config>
```

5.5 Creating a Portlet Producer Application

A portlet producer application is configured to build standards-based (JSR286) and Oracle PDK-Java portlets. This section explains how to create a portlet application

using the Portlet Producer Application template. See also Chapter 58, "Overview of Portlets."

For information about extending an existing application to enable portlet creation, see Section 6.2, "Extending Non-WebCenter Portal: Framework Applications to Include WebCenter Capabilities."

This section includes these topics:

- Section 5.5.1, "How to Create a Portlet Application Using a Template"
- Section 5.5.2, "What Happens When You Use a Portlet Producer Application Template"

5.5.1 How to Create a Portlet Application Using a Template

The Portlet Producer Application template consists of a single project, *Portlets*. Use the Portlets project to create JSR 286 (standards-based) and Oracle PDK-Java portlets.

To create an application using the Portlet Producer Application template:

1. Access the application creation wizard in any of the following ways:
   - From the **File** menu, choose **New**. In the New Gallery dialog, expand **Applications**, select **Portlet Producer Application**, and click **OK**.
   - From the Application Navigator, select **New Application**.
   - If you have an application open, then in the Application Navigator open the application name drop down menu and select **New Application**.
   - In the Application Navigator, right-click the application name and choose **New**. In the New Gallery dialog, expand **Applications**, select **Portlet Producer Application**, and click **OK**.
2. In the Name your application page of the creation wizard, enter a name for the application in the **Application Name** field.
3. In the **Directory** field, accept the default path or enter a path to the directory where the application should be stored.

   For example:

   ```
   C:\JDeveloper\mywork\Application1
   ```

   Optionally, click the **Browse** button to navigate to the desired directory.
4. If required, in the **Application Package Prefix** field, enter a prefix to use for packages to be created within this application.
5. From the **Application Template** list, select **Portlet Producer Application**.

   Note: The Application Template list appears only when you invoke the wizard by clicking **New Application** in the Application Navigator as described in step 1.
6. Click **Finish** to create the portlet application with default project configurations.

5.5.2 What Happens When You Use a Portlet Producer Application Template

Using the Portlet Producer Application template generates the default *Portlets* project with the Java, JSP and Servlets, and Portlet technology scopes. It limits the types of

options that are exposed in the JDeveloper user interface to those appropriate for creating portlets.

When you right-click this new project, options to create JSR 286 and Oracle PDK-Java portlets are provided.

5.6 How Is a Framework application Security Configured by Default?

By default, a WebCenter Portal: Framework application is configured with ADF security. A default username and password (weblogic/weblogic1) are created automatically for you, and you can use this username/password combination for testing purposes immediately.

Default login and logout pages are also provided with the WebCenter Portal: Framework Application template.

For more information on ADF security, see Chapter 69, "Securing Your WebCenter Portal: Framework Application." See also the section on *Enforcing ADF Security in a Fusion Web Application* in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6 Advanced Configuration Options

This chapter discusses techniques for manually configuring and extending a WebCenter Portal: Framework application. This chapter includes the following sections:

- Section 6.1, "Manually Configuring a Framework Application With Technology Scopes"
- Section 6.2, "Extending Non-WebCenter Portal: Framework Applications to Include WebCenter Capabilities"
- Section 6.3, "Creating a Project by Importing a WAR File"

6.1 Manually Configuring a Framework Application With Technology Scopes

This section discusses how to manually configure a WebCenter Portal: Framework application by adding or removing technology scopes.

- Section 6.1.1, "What are Technology Scopes?"

  Section 6.1.2, "Adding and Removing Technology Scopes"
- Section 6.1.3, "What Are the Default Technology Scopes for a Framework Application?"
- Section 6.1.4, "Which Technology Scopes Are WebCenter Portal: Framework Application-Specific?"
- Section 6.1.5, "Can I Create a WebCenter Portal: Framework Application by Adding the Oracle WebCenter Portal: Framework Technology Scope Later?"
- Section 6.1.6, "Migrating a WebCenter PS2 or Earlier Application to a WebCenter Portal: Framework Application"

6.1.1 What are Technology Scopes?

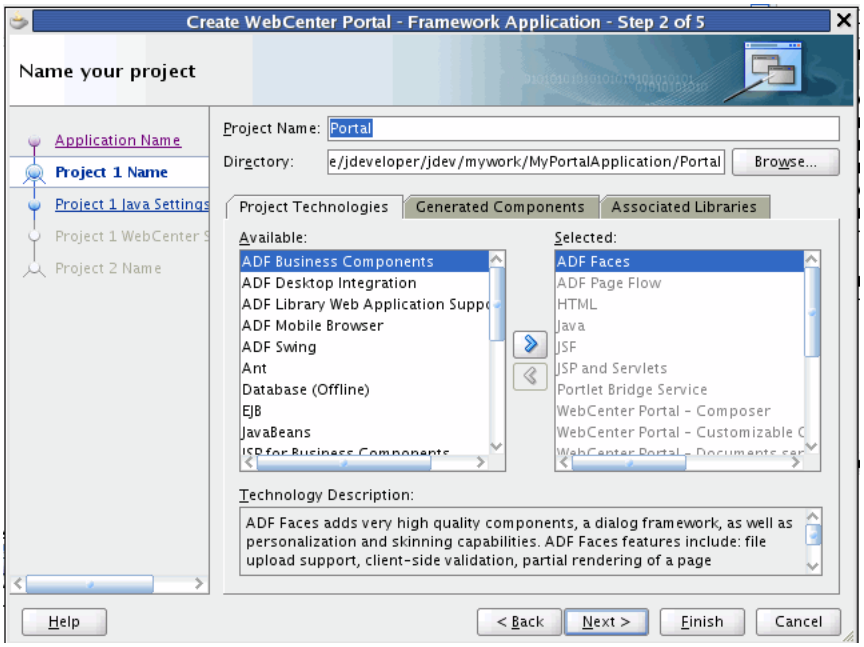
Framework applications are comprised of a collection of project technologies and features or *technology scopes*. Each scope has associated libraries that provide specific features to the application. You can manually add or remove technology scopes to manually configure your application. For example, you may wish to remove some components from your Framework application that you know you will never use.

6.1.2 Adding and Removing Technology Scopes

You can add or remove technology scopes (and their associated libraries) in the following ways.

First, you can modify the defaults when you create a new application. Figure 6–1 shows the list of available and selected technology scopes presented in the Create WebCenter Portal – Framework Application wizard. The scopes in the Selected column indicate the default set of technology scopes that comprise a framework application. For more information on this wizard, see Section 5.2, "Creating a Framework Application."

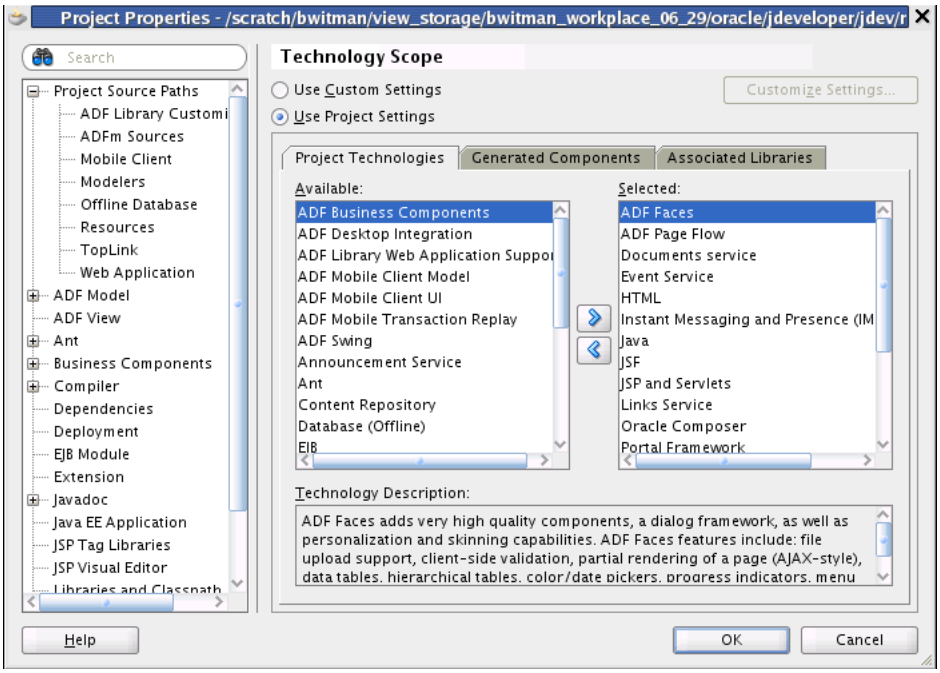
Figure 6–1 Create WebCenter Portal: Framework Application Wizard Step 2 of 5



You can also adjust technology scopes in the Project Properties dialog. Select **Project Properties** from the Application menu. Then, select **Technology Scope** from the tree. The dialog lets you move project technologies in and out of the project, as shown in Figure 6–2.

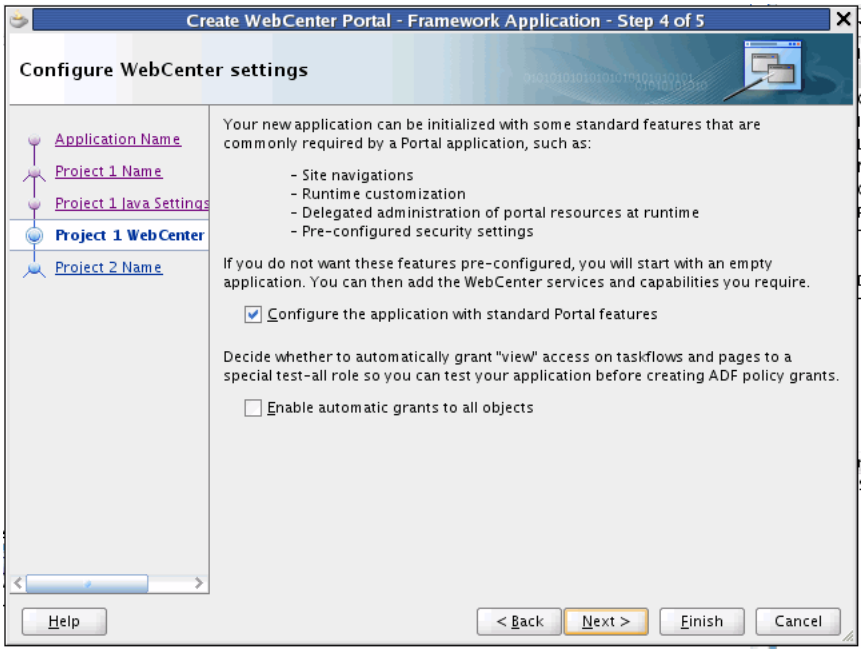
> **Tip:** Click on a scope in the dialog to display a short description at the bottom of the dialog.

Figure 6–2 The Technology Scope Dialog



When you create a Framework application using the Create WebCenter Portal – Framework Application wizard, you have one other opportunity to determine the default configuration of the application that is created, and its technology scopes. The final screen of the wizard includes a checkbox labeled **Configure the application with standard Portal features**, shown in Figure 6–3.

Figure 6–3 Create WebCenter Portal: Framework Application Wizard Step 4 of 5



Checked by default, this option tells the wizard to add the Oracle WebCenter Portal: Framework features, like navigations, page hierarchies, delegated administration, and so on, to the application. If you uncheck this box, your application will not include

these features. Essentially, you will create an application that is functionally equivalent to a WebCenter PS2 (WebCenter 11g Release 1 11.1.1.2.0) application.

6.1.3 What Are the Default Technology Scopes for a Framework Application?

Table 6–1 lists the *default* technology scopes and libraries that are included with applications created with the WebCenter Portal – Framework Application template. The **Oracle WebCenter Portal: Framework** technology scope is the only scope that is portal-application specific. This scope provides navigations, page hierarchies, delegated administration, and the rest of the portal features described in Chapter 9, "Understanding Framework Applications."

> **Note:** The Oracle WebCenter Portal: Framework technology scope is added by default to an application created with the Framework application template. The Framework application creation wizard gives you a chance to omit this technology scope if you unselect the **Configure the application with standard Portal features** checkbox in the Framework application creation wizard (Figure 6–3). In this case, you will create an application that is functionally equivalent to a WebCenter PS2 (WebCenter 11g Release 1 11.1.1.2.0) application.

Table 6–1 Default Technology Scopes and Associated Libraries

Default Technology Scopes	Associated Libraries
ADF Faces	JSF Core
ADF Page Flow	JSF HTML
Documents Service	ADF Faces Runtime 11
Events Service	ADF Common Runtime
HTML	ADF Web Runtime
Instant Messaging and Presence (IMP) Service	MDS Runtime
Java	MDS Runtime Dependencies
JSF	Commons Beanutils 1.6.1
JSP and Servlets	Commons Logging 1.0.3
Links Service	Commons Collections 2.1
Oracle WebCenter Portal's Composer	ADF Page Flow Runtime
WebCenter Portal – Framework	ADF Controller Runtime
Portlet Bridge Service	JSP Runtime
Portlet View	
Smart Tag	
Tags Service	
WebCenter Portal's Customizable Components	
WebCenter Portal: Services	
XML	

6.1.4 Which Technology Scopes Are WebCenter Portal: Framework Application-Specific?

When the Configure the application with standard Portal features checkbox is selected in the Framework application creation wizard (Figure 6–3), the Oracle WebCenter Portal: Framework technology scope is added to the application in addition to the rest of the application technology scopes. If you were to unselect this checkbox, the Oracle WebCenter Portal: Framework technology scope would not be added.

6.1.5 Can I Create a WebCenter Portal: Framework Application by Adding the Oracle WebCenter Portal: Framework Technology Scope Later?

This case is not supported. If you want to create a Framework application, you must start with the WebCenter Portal: Framework Application template. For details, see Section 5, "Creating Framework and Portlet Producer Applications."

One possible solution is to create a new Framework application, then turn your existing Framework application into an ADF Library. Then, include the ADF Library in the WebCenter Portal: Framework Application.

6.1.6 Migrating a WebCenter PS2 or Earlier Application to a WebCenter Portal: Framework Application

The recommended best practice for migration is to create a new (PS3) Framework application using the WebCenter Portal: Framework Application template and then manually migrate the content, configurations, and logic from your old PS2 application to the new framework application. If it is not possible to follow this recommended practice, then follow the procedures described in Appendix H, "Manually Migrating a Framework Application."

> **Tip:** Another possible solution is to create a new Framework application, and then turn your existing Framework application into an ADF Library. Then, include the ADF Library in the Framework application.

6.2 Extending Non-WebCenter Portal: Framework Applications to Include WebCenter Capabilities

If your business requirement is to create a non-Framework application, say a Fusion web application, but you want to also include WebCenter capabilities in that application, you can do so by adding the required technology scopes and libraries. For example, you could add features like web services, task flows, and search to the application.

> **Note:** You cannot create a Framework application simply by adding the Oracle WebCenter Portal: Framework technology scope to a non-portal application. This use case is not supported, as explained in Section 6.1.5, "Can I Create a WebCenter Portal: Framework Application by Adding the Oracle WebCenter Portal: Framework Technology Scope Later?."

The technology scopes limit user interface options to those appropriate to the type of application project you are building. The libraries provide the components and elements useful in constructing application content. You can extend an existing

application to integrate content from various repositories, create portlets, or consume portlets and WebCenter Portal services. You can extend ADF applications from the current release or from a 10.1.x.x release.

While extending your application, you may decide to create projects that are optimized for specific WebCenter capabilities. This section steps you through the process of manually creating projects like those provided with the WebCenter templates and adding the required technology scopes and libraries to projects.

To enable WebCenter capabilities in an application:

1. Open your application.

   Click **Yes** if you are prompted to migrate your application settings.

   > **Note:** If you migrated a 10.1.3.x Framework application, then the application continues to have the Model, ViewController, and Portlets projects.

2. Optionally, create new projects for defining the application data model, consuming content, or creating portlets:
   a. In the Application Navigator, right-click the application and select **New Project** from the context menu.
   b. In the New Gallery, expand **General**, select **Projects**, then **Generic Project**, and click **OK**.
   c. In the Create Generic Project wizard, in the **Project Name** field, enter a name for the project.

      For example `View`.

   d. In the **Directory** field, specify the directory path for storing the project or accept the default path.
   e. Click **Finish** to create the new project.
3. Right-click a project and select **Project Properties**.
4. In the left side of the dialog, select **Technology Scope**.
5. In the Project Technologies part of the dialog, add the required technologies and libraries depending on which WebCenter-related capabilities you want to enable in the selected project. See Table 6–1 for a complete list.
6. Save your work.

6.3 Creating a Project by Importing a WAR File

Follow this procedure if you have an existing WAR file you want to import into your application environment through JDeveloper. Once the project is imported, you can configure it with WebCenter Portal features manually, as described in Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."

> **Note:** Connection information is stored in the application-level EAR file and is not automatically imported with a project-level WAR file. Therefore, when you import a project-level WAR file, you must reconfigure any connections upon which the application relies.

To import a WAR file into JDeveloper:

1. Right-click your application in the Application Navigator, and choose **New Project** from the context menu.
2. In the New Gallery, expand **General**, select **Projects**, then **Project from War File**, and click **OK**.
3. Follow the wizard instructions to complete creating the project.
4. Right-click the newly created project in the Application Navigator, and select **Project Properties** from the context menu.
5. Add the required technology scopes and libraries depending on which WebCenter-related capabilities you want to enable in the selected project. For more information, see Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."
6. Click **OK** to close the Project Properties dialog.

7

Preparing Your Application for WebCenter Portal Services

This chapter describes how to prepare your application to consume services in Oracle WebCenter Portal.

This chapter includes the following sections:

- Section 7.1, "Understanding WebCenter Portal Services Technologies"
- Section 7.2, "Preparing Your Framework Application to Consume Services"
- Section 7.3, "Extending Your Framework Application with Custom Components"
- Section 7.4, "Configuring General Settings for Your Services"

7.1 Understanding WebCenter Portal Services Technologies

Oracle WebCenter Portal Services enrich existing portals and Web sites with enterprise 2.0 capabilities, including social computing services, personal productivity services, online awareness and communications, content integration, and Web analytics.

See Also:

- Section 2.4, "Introducing Oracle WebCenter Portal: Services" for conceptual information about services
- Chapter 3, "Preparing Your Development Environment" for fundamental information about Framework applications, such as how to create an application

This section summarizes the key technologies available in services. This can help you determine which services to implement and the best way to customize your user interface.

This section includes the following subsections:

- Section 7.1.1, "Understanding WebCenter Portal Horizontal Services"
- Section 7.1.2, "Understanding WebCenter Portal APIs"
- Section 7.1.3, "Using WebCenter Portal Data Controls"

7.1.1 Understanding WebCenter Portal Horizontal Services

Some services in WebCenter Portal are considered *horizontal* services in that they interact with other services across an application.

Table 7–1 lists service integration in Oracle WebCenter Portal: Framework.

Table 7–1 Service Integration in WebCenter Portal

WebCenter Portal Services	RSS	Search	Tags	Links	Comments and Likes	Activity Stream	Activity Graph	Notifications	Analytics
Activity Graph						X	n/a		X
Announcements	X	X		X		X		X	
Blogs		X	X	X	X	X	X	X	X
Discussions	X	X	X	X	X (replies on topics)	X	X (replies)	X	
Documents, including wikis		X	X	X	X	X	X	X	X
Events		X partial		X		X		X	
Instant Messaging and Presence (IMP)								X	
Lists	X	X	X	X		X		X	
Mail								X	
Notes		X		X					
People Connections		X partial	X partial		X (Activity Stream, Message Board)	X (Message Board, Profiles, Feedback	X (Message Board)	X	X
Polls									
Recent Activities	X								
Search		n/a	X			X partial			X
Tags		X	n/a			X			

Notes: Notes, space Events and Favorites are available *only* in WebCenter Portal: Spaces applications.

The Search service (listed in the table) uses WebCenter Portal search adapters to search each available service. However, large-scale implementations should be configured to use Oracle SES search for best performance. Oracle SES searches applications for the following resources:

- Documents, including wikis and blogs
- Announcements and Discussions
- WebCenter Portal: Spaces applications only: spaces, lists, pages, and people resources in Spaces

For more information, see Chapter 45, "Integrating the Search Service."

7.1.2 Understanding WebCenter Portal APIs

This section describes the APIs available for WebCenter Portal services.

Note the following considerations when multiple customization options are available:

- Use REST APIs to build a custom client for accessing WebCenter Portal: Spaces; for example, an iPhone application that interacts with Spaces.
- REST APIs may provide the fastest and easiest way to customize the user interface and develop custom components. Consider using data controls or native APIs to build a custom user interface for accessing Spaces.
- Use data controls or native APIs to build a custom portal or composite Framework application.
- Use data controls or native APIs if you are proficient with Java, JDeveloper, and ADF. To use other languages, especially JavaScript, use REST APIs.

Table 7–2 lists services with available APIs. For services that use a 3rd-party back end, it includes available back end APIs.

Table 7–2 Supported Technologies for Services

WebCenter Portal Service	Data Controls	Java APIs	REST APIs	Back End APIs
Activity Graph	Section 46.3.1, "Using the Activity Graph Data Control"	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Section 46.3.2, "Using the Activity Graph Service REST APIs"	
Analytics	Section 47.3.1, "Using SQL Data Controls"	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*		
Announcements				Web Services with Oracle WebCenter Portal's Discussions Server (requires configuration, because WebCenter Portal is set up with custom authentication) See the Jive Forums documentation on the Oracle Fusion Middleware documentation library (in the WebCenter Portal product area)
Comments & Likes		*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Section 42.1, "Activity Stream REST APIs"	
Discussions			Section 34.3.8, "Using the Discussions Service REST APIs"	Web Services with Oracle WebCenter Portal's Discussions Server (requires configuration, because WebCenter Portal is set up with custom authentication) See the Jive Forums documentation on the Oracle Fusion Middleware documentation library (in the WebCenter Portal product area)

Table 7–2 (Cont.) Supported Technologies for Services

WebCenter Portal Service	Data Controls	Java APIs	REST APIs	Back End APIs
Documents (with wiki and blogs)	Chapter 28, "Working with Content Data Controls"	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Oracle Fusion Middleware Content Management REST Service Developer's Guide	See *Oracle Fusion Middleware Remote Intradoc Client (RIDC) Java API Reference* *Oracle Fusion Middleware Services Reference Guide for Oracle Universal Content Management*
Events			Section 52.3, "Using the Events Service REST APIs"	Web Services with Microsoft Exchange
General Settings		*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*		
Links			Section 43.3.1, "Using the Links Service REST APIs"	
Lists			Section 48.4, "Using the Lists Service REST APIs"	
Notifications	Section 49.3.2, "Using Notifications Data Controls"	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*		
Pagelet Producer			Section 63.2.2.2, "Accessing Pagelets Using REST"	
People Connections	Section 41.1, "Using People Connections Data Controls"	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Chapter 42, "People Connections Service REST APIs" (for Profile and Activity Stream)	
Personalization		*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal* (for developing custom Providers and Locators)	Section 67.4, "Extending Personalization"	
Polls	Section 37.3.4, "Using the Polls Service Data Controls"			

Table 7–2 (Cont.) Supported Technologies for Services

WebCenter Portal Service	Data Controls	Java APIs	REST APIs	Back End APIs
Resource Action Handler Framework		*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*		
Search	Section 45.3.8, "Using the Search Service Data Control" **Note**: Only Oracle SES search supports the search data control.	*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Section 45.3.7, "Using the Search Service REST APIs" **Note**: Only Oracle SES search supports the search REST APIs.	Web Services and Java APIs with Oracle Secure Enterprise Search
Tags		*Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*	Section 44.3.2, "Using the Tags Service REST APIs"	

See Also: *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal* for more information about Expression Language APIs

7.1.3 Using WebCenter Portal Data Controls

Several services provide data controls for building a customized user interface with a Framework application or task flow. Deploying this task flow into an ADF library allows for a portable consumption of the task flow; for example, you could add it to a Spaces application Resource Catalog (Composer) to build site templates.

You do not need to integrate a service before you can use its data control to edit the user interface. An application is configured for the respective service when one of its data controls or task flows is used.

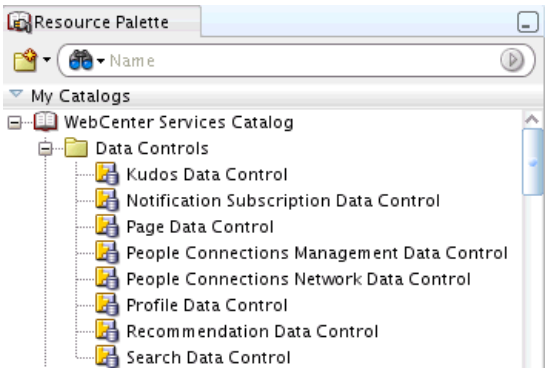
Note: Oracle ADF architecture provides data controls for the user interface to understand the structure of your data. Metadata describes data collections, properties, methods, and types. When you drag and drop attributes, collections, and methods onto a page, JDeveloper automatically creates the bindings from the page to the associated services.

See *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for detailed information about data controls.

To use a built-in data control:

1. In JDeveloper, create a WebCenter Portal Framework application with any connections required for the services.
2. In the Resource Palette, expand My Catalogs, WebCenter Portal - Services Catalog, and Data Controls.
3. Right-click the data control and choose **Add to Project**.

Figure 7–1 Data Controls in the Resource Palette



4. Drag and drop methods and properties from the data control to a ADF JSF page.

Note: There are two other ways to add data controls to your application:

- From the Resource Palette, choose IDE connections, and navigate to the data control. Right-click the data control, and select **Add to Project**.
- From Project Properties, choose Libraries and Classpath, and click the **Add Library** button. With this approach, you must know the JDeveloper library that contains the data control you want.

7.2 Preparing Your Framework Application to Consume Services

This section describes the steps you must take to prepare your application to use WebCenter Portal Web services. It includes the following subsections:

- Section 7.2.1, "How to Prepare Your Application to Consume Services"
- Section 7.2.2, "Setting Up a Database Connection"
- Section 7.2.3, "Setting Up an External Application Connection"

7.2.1 How to Prepare Your Application to Consume Services

You can configure any application to include a service. When you create an application in JDeveloper, you can choose to base the application on a template. Although not a requirement, WebCenter Portal's Framework application template makes all the appropriate connection wizards and tag libraries readily visible and available in the New Gallery and Component palette. When you consume a service task flow or component, the necessary libraries are automatically added to the project.

Depending upon the service you plan to consume, your application must meet certain prerequisites. For example, if the service must know the identity of users, then your application must provide some level of security with user authentication.

This section includes the following subsections:

- Section 7.2.1.1, "Implementing Security for Services"
- Section 7.2.1.2, "Setting Up SSL-Protected Connections for Services"

7.2.1.1 Implementing Security for Services

Some services must know the identity of the user (for example, the Search service needs the user's identity for saving searches). For these services, you must at least configure your application to authenticate users such that they have distinct identities for the purposes of user customization and preferences.

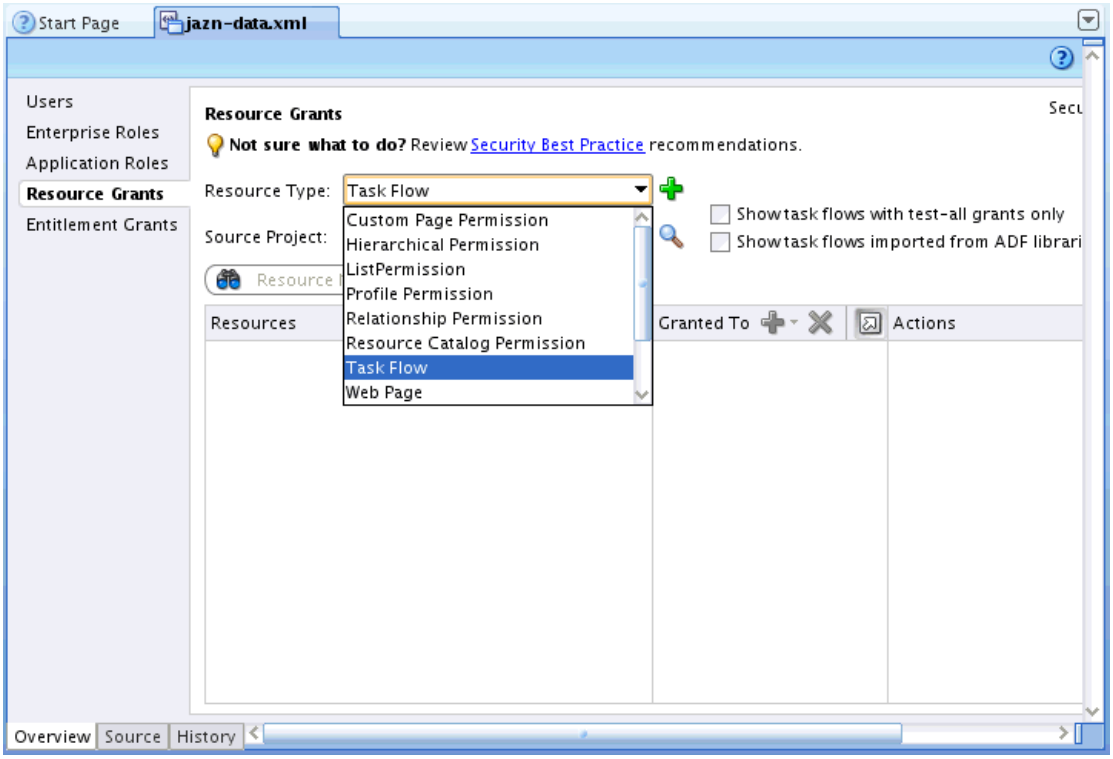
ADF security is configured by default if you created your application using WebCenter Portal's Framework application template.

See Also:

- Section 69.3, "Configuring ADF Security" for details on how to implement a basic security solution for your Framework application
- Chapter 69, "Securing Your WebCenter Portal: Framework Application" for details on how to implement a complete security solution for your Framework application

After you configure ADF security for your application, you can open the `jazn-data.xml` file and modify your sample user's privileges for each task flow. To open the ADF Security Policies Editor, locate the file in the Application Resources panel and double-click its name, or select Application - Secure - Resource Grants (Figure 7–2). The Resource Type dropdown controls the set of grants that are shown in the table. You can show only the grants for a particular permission type by selecting it from the list.

Figure 7–2 ADF Security Policies in the jazn-data.xml File



7.2.1.2 Setting Up SSL-Protected Connections for Services

If you are setting up a connection in JDeveloper for a service, and the connection is being made to an SSL-protected endpoint (with a valid certificate from a trusted certificate authority) then you need prepare your environment accordingly.

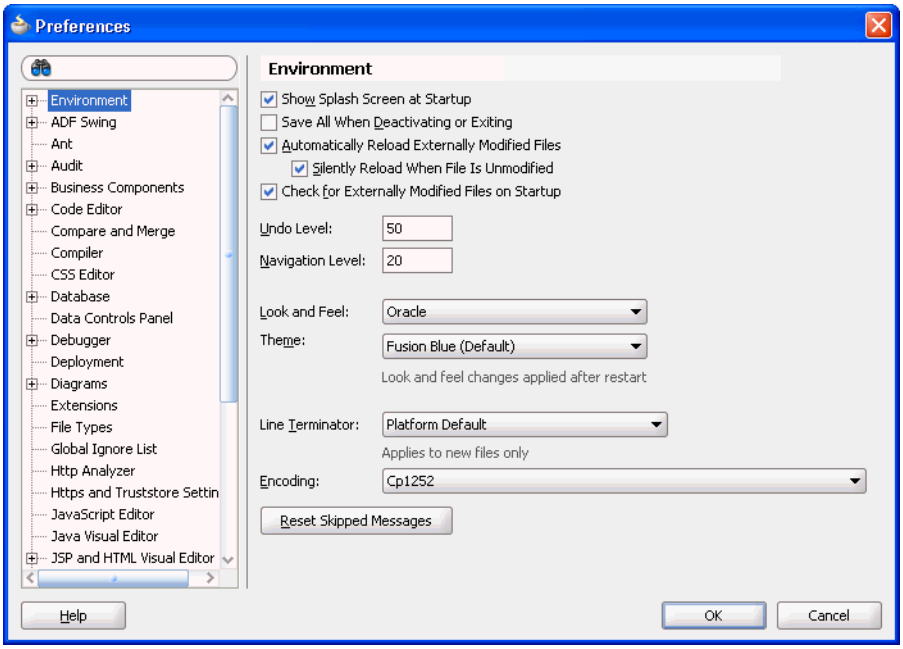
> **Note:** This Preferences setting change is required only for connections made using Integrated WLS as the default server. If you are deploying to a Managed Server, this settings change is not required.

To set preferences for SSL-protected service connections:

1. From the JDeveloper tool bar, select **Tools > Preferences**.

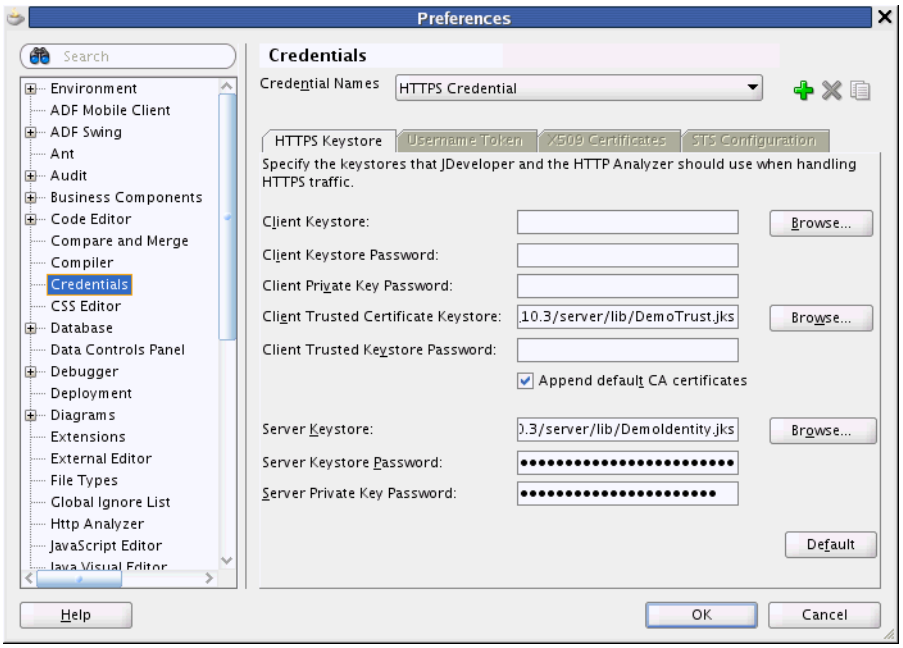
   The Preferences dialog displays (Figure 7–3).

Figure 7–3 Preferences Dialog



2. On the Preferences dialog, click **Credentials** (Figure 7–4).

Figure 7–4 Credentials Pane



3. Change the value of **Client Trusted Certificate Keystore** to:

   `<JAVA_HOME>/jre/lib/security/cacerts`

   where `<JAVA_HOME>` is the location of the Java home directory.

4. Click **OK**.

7.2.2 Setting Up a Database Connection

Many services require a connection to a database schema where relevant information is stored. For example, with the Links service, relationship mapping information, such as what object is linked to what other object, is stored in the database.

Table 7–3 lists the services that require a database connection.

Table 7–3 Associated Data Source

Service	Data Source
Activity Graph	Activities
Activity Stream	WebCenter Portal
Analytics	Activities
Blogs	WebCenter Portal
Comments	WebCenter Portal
Documents (including wikis and blogs) that want to include Comments and Activity Stream	WebCenter Portal
Links	WebCenter Portal
Lists	WebCenter Portal
Oracle Portal Adapter	PORTAL (name of the Oracle Portal schema)
People Connections	WebCenter Portal

Table 7–3 (Cont.) Associated Data Source

Service	Data Source
Polls	WebCenter Portal
Tags	WebCenter Portal

See Also:

- Section 3.5, "Installing Database Schemas" for information about installing the schemas
- "Deploying the Application to a WebLogic Managed Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* for data source considerations when deploying your application to a production environment

To create the database connection:

1. In the Application Navigator, expand the **Application Resources** panel.
2. Right-click **Connections**, then click **Database.**
3. Enter the following information for your database connection:
   - **Connection Name**: `webcenter/CustomPortal` or `activities/CustomPortal`

     Note: Oracle recommends that you use these names for ease of deployment to a Framework managed server.

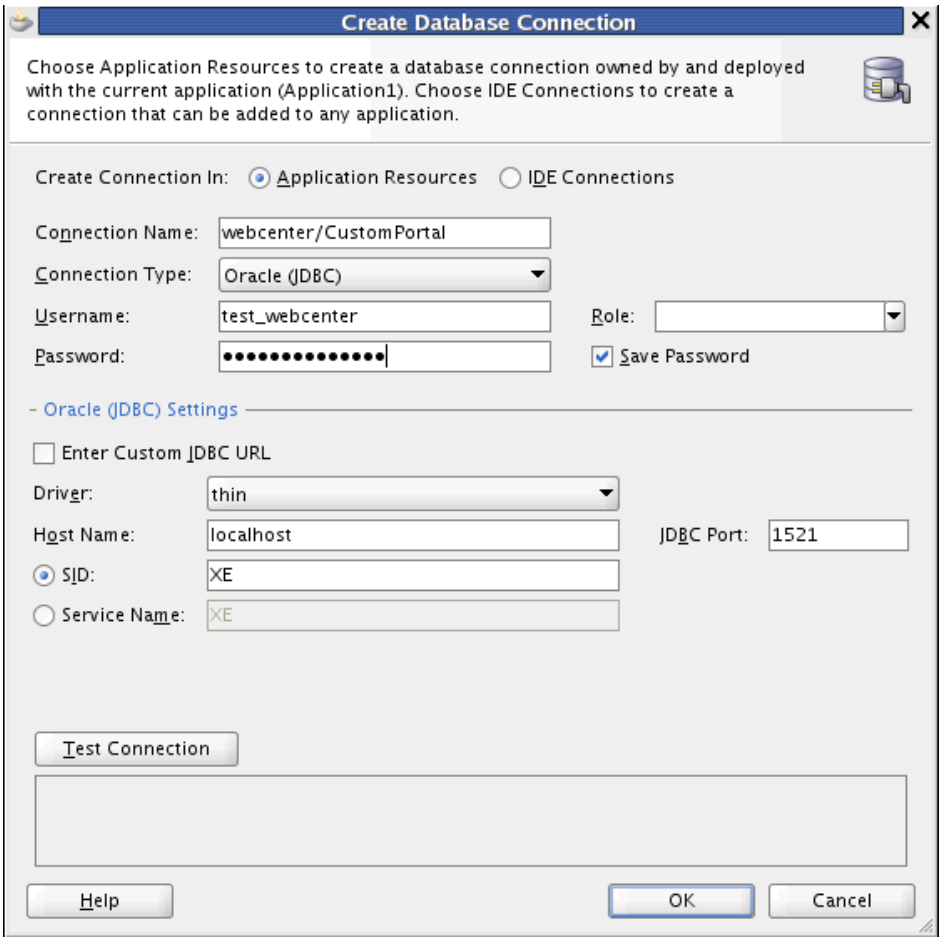
     - If your application contains task flows that use the WebCenter Portal schema, then name the database connection `webcenter/CustomPortal`.
     - If the application contains task flows that use the Activities schema, then name the database connection `activities/CustomPortal`.

   - **Connection Type**: `Oracle (JDBC)`
   - **Username:** `username`
   - **Password**: `password`
   - **Host**: *host where you install the WebCenter Portal schema*; for example, `localhost`
   - **JDBC Port:** *port*; for example, `1521`
   - **SID**: *system identifier for the database with the same JDBC*; for example, `ORCL`

     Note: If the **Save Password** checkbox is not selected, then when deploying from JDeveloper to a managed server or the Integrated WebLogic Server, you must manually create the data source after deployment.

     For detailed information on how to create a JDBC data source for Oracle WebLogic Server, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

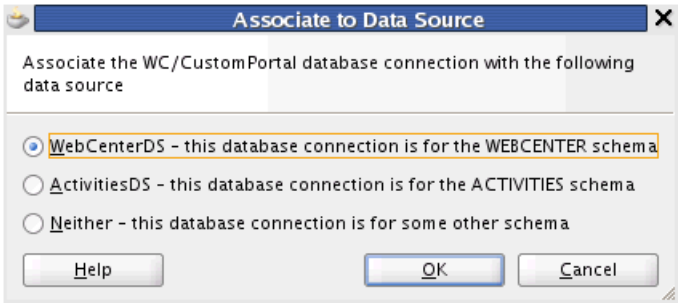
Figure 7–5 Database Connection



4. Click **OK**.
5. In the Associate to Data Source dialog, select the appropriate schema.
   - If you named the connection `webcenter/CustomPortal`, then select to associate this connection to the WebCenter Portal schema.
   - If you named the connection `activities/CustomPortal`, then select to associate this connection to the Activities schema.

   Click **OK** (Figure 7–6).

Figure 7–6 Associating to Data Source from the Connection Wizard



The data source association applies separately to each project in the workspace. If there is more than one WebCenter Portal project in the workspace, then the dialog displays a dropdown list allowing the user to choose which project to configure.

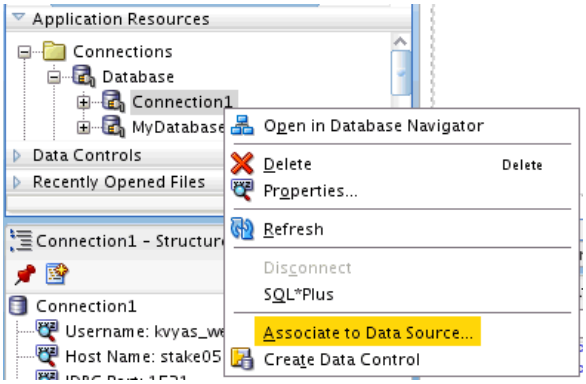
Note: While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control.

See *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* for more information.

7.2.2.1 Associating Existing Database Connections

For existing database connections, you can associate data sources by right-clicking the connection and selecting **Associate to Data Source** (Figure 7–7).

Figure 7–7 Associating to Data Source for Existing Connections



7.2.3 Setting Up an External Application Connection

When a service interacts with an application that handles its own authentication, you can associate that application with an external application definition to allow for credential provisioning.

The following services permit the use of an external application to connect with the service and define authentication for it:

- Documents
- Events
- Instant Messaging and Presence
- Mail
- RSS Viewer (when using a secured RSS feed)

See Also:

- Chapter 30, "Integrating the Documents Service," Chapter 35, "Integrating the Instant Messaging and Presence Service," Chapter 36, "Integrating the Mail Service," and Chapter 53, "Integrating the RSS Service" for more information about setting up the external application for a service
- Section 69.13, "Working with External Applications" for more information about external applications in general

7.3 Extending Your Framework Application with Custom Components

This section introduces the Resource Action Handling Framework and describes how to register a resource viewer. It includes the following subsections:

- Section 7.3.1, "Introducing the Resource Action Handling Framework"
- Section 7.3.2, "Registering a Resource Viewer"
- Section 7.3.3, "Creating a Resource Authorizer Class"
- Section 7.3.4, "Specifying the View Handler and Navigation Behavior"

7.3.1 Introducing the Resource Action Handling Framework

Custom components, just like out-of-the-box services, manage and own resources. As such, they must make declarations that enable their resources to be accessible to the services that invoke other services (for example, the Search, Tags, Links, and Recent Activities services). WebCenter Portal provides a *Resource Action Handling framework* for custom components and services that expose resources to be viewed, searched, and tagged.

For example, the Resource Action Handling framework enables the Search service to look up and follow the mechanism to render a resource for any given service. Other facilities of the Resource Action Handling framework allow for authorization of resources to be declared and enforced when access is made.

The following resources invoke resource viewers using the Resource Action Handling framework:

- Search
- Tags
- Links
- Wikis and Blogs
- Recent Activities
- People Connections
  - Activity Stream (invoking itself)
  - Connections (invoking itself)
  - Feedback (invoking itself)
  - Profile (invoking Tags, Tagged Items, and itself)
  - Message Board (invoking itself)
- External links pointing into WebCenter Portal (from, for example, mail, notifications, documents, RSS feeds, or REST XML)

The following resources own task flows that can be invoked by the Resource Action Handling framework.

- Announcements
- Discussions
- Documents, including wikis and blogs
- Events
- Lists
- Mail
- Notes
- Page
- People Connections
  - Activity Stream - Main View
  - Connections
  - Feedback
  - Profile
  - Profile Gallery
  - Message Board
- Spaces (URL rewriter that maps to `/webcenter/spaces/Space1Name`)
- Tags - Tag Center

> **See Also:** *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal* for more information on the Resource Action Handling and rewriter interfaces

7.3.2 Registering a Resource Viewer

You must register a resource viewer to enable custom resources to be rendered using Search or Tags, or to make the resources linkable to and from each other.

In the `service-definition.xml` file, you can define a resource viewer for any service to render that service's resources. The viewer can be a task flow or a URL rewriter acting on the resource ID.

For example, suppose a discussion forum message is found through the Search service. Clicking the result link launches the resource viewer that was declared for the Discussions service (by looking up the `service-definition.xml` of the Discussions service). By default, the target is rendered in a dialog.

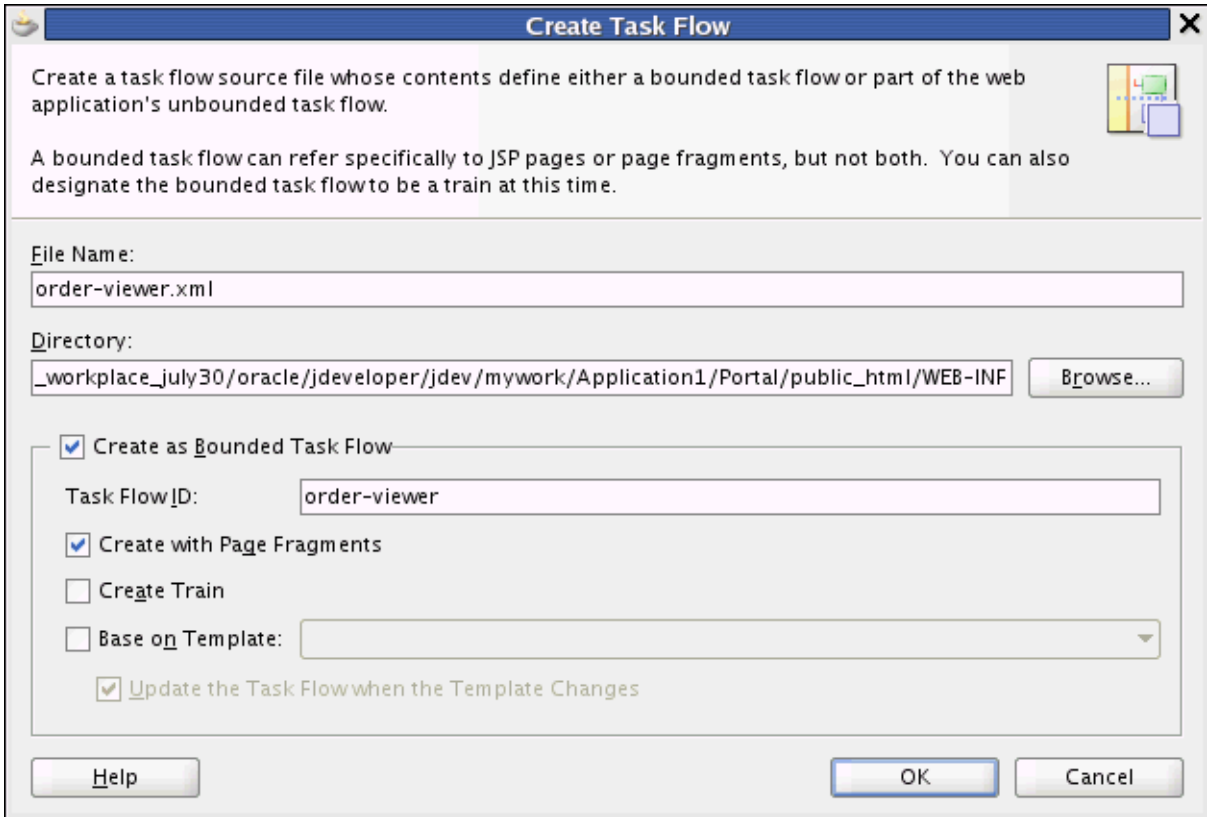
You can declare a resource mini-view to specify a resource viewer that is used for any inline popups displayed for this service. If the mini-view is not specified, then the resource view is used for inline popups.

Note: A sample `service-definition.xml` is included with Framework applications. This file enables developers to declare their resource viewers and their implementations of interfaces for interacting with the Links and Search services; for example, the QueryManager interface implementation.

To register a custom resource viewer:

1. In the Application Navigator, right-click the **Page Flows** node for the project and select **New** from the context menu.
2. Under **Categories**, select **JSF**, then **ADF Task Flow**, and click **OK**.
3. In the Create Task Flow dialog, change **File Name**, for example, to `order-viewer.xml`. The Task Flow ID should automatically update to `order-viewer` (Figure 7–8).

Figure 7–8 Create Task Flow Dialog for Resource Viewer



4. Click **OK**.

   Ensure that the `order-viewer.xml` file is created under `..\public_html\WEB-INF`.
5. From the Component Palette, ensure that ADF Task Flow is selected and drag **View** onto the task flow to create a page fragment.
6. Rename the page fragment to `order-viewer`, and click in an open area of the canvas to accept the change.
7. Define a JSFF file for your page fragment.
8. Double-click the page fragment.
9. Click **OK** in the Create New JSF Page Fragment dialog.
10. Go to the Source view in the editor and paste an `af:outputText` tag inside of the `jsp:root` element.

Example 7–1 provides a very simple resource viewer. You could create a much more complex viewer.

Example 7–1 Simple Resource Viewer

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <af:outputText value="Rendering #{pageFlowScope.resourceId}"
                 inlineStyle="font-size:5"/>
</jsp:root>
```

11. Add a `resourceId` input parameter to the task flow, and set the value to `#{pageFlowScope.resourceId}`.
12. After the task flow is defined, you must register it in the `service-definition.xml` file. In the Application Navigator, expand **Application Resources**, **Descriptors** and **ADF META-INF**.
13. Double-click `service-definition.xml` to open it in the editor.
14. In the file, you see a service definition that has been commented out. Uncomment it by removing the `<!--` and `-->` tags around it.
15. Set the `id` attribute to the service identifier.

    For example, the service identifier used when defining the Tagging Button could be `mycompany.myproduct.myapp.mycomponent`.
16. Set the `taskFlowId` in the `<resource-view>` element to `/WEB-INF/order-viewer.xml#resource-viewer`.

    This identifier is the task flow path and file name concatenated with the task flow identifier declared in the Create Task Flow dialog. You can omit the remaining tags.
17. The `<resource-view>` declaration has a class specification for the Resource Authorizer. This provides various ways to specify the resource view and determine which tagged objects are visible (and searchable) to users.

Example 7–2 Specifying a Task Flow ID in a resource-view Declaration

```
<resource-view taskFlowId="/somepath/somefile.xml#someId"
               authorizerClass="some.service.id.Authorizer1"/>
```

The Search, Tags, and Discussions services use this type of declaration. Instead of having a view ID (JSPX page), the service supplies a task flow to render the resource.

- The attribute `taskFlowId` signifies that the renderer is a task flow.
- A task flow is used to render the resource specified by the resource ID.
- The task flow honors a `resourceId` input parameter and uses that to render the resource.

Example 7–3 Specifying a URL Rewriter Class Name (1)

```
<resource-view
urlRewriterClass="oracle.webcenter.framework.resource.IdentityResourceUrlRewriter"
               authorizerClass="some.service.id.Authorizer5"/>
```

or

Example 7–4 Specifying a URL Rewriter Class Name (2)

```
<resource-view
urlRewriterClass="oracle.webcenter.framework.resource.MessageFormatResourceUrlRewr
iter"
               authorizerClass="some.service.id.Authorizer4">
  <parameters>
    <!-- This starts from the context root which would be /webcenter -->
    <parameter name="message" value="/content/conn/UCM/path/{0}"/>
  </parameters>
</resource-view>
```

or

Example 7–5 Specifying a URL Rewriter Class Name (3)

```
<resource-view
urlRewriterClass="oracle.webcenter.framework.resource.MyResourceUrlRewriter"
               authorizerClass="some.service.id.Authorizer6">
  <parameters>
    <parameter name="myParam" value="myValue"/>
  </parameters>
</resource-view>
```

Some services use a URL rewriter to treat the resource ID as a URL and rewrite the URL in an appropriate fashion. A URL rewriter changes the resource ID into a URL that can be launched with a `goLink`. Two `ResourceUrlRewriters` are provided:

- `oracle.webcenter.framework.resource.IdentityResourceUrlRewriter` is for Search with Oracle SES. It takes the `resourceId` value as a URL as is.
- `oracle.webcenter.framework.resource.MessageFormatResourceUrlRewriter` is for Documents. It takes in a `message` parameter (in the `service-definition`). The `message` parameter contains a {0}, which is replaced with the value of the `resourceId`.

The `urlExternal` parameter applies to all URL rewriters. This parameter's value determines whether the JSPX that consumes a resource action handler link should open a new window.

Some resources, like pages and wikis, can have no resource viewer but may allow dubbing their resource IDs as page view IDs (Example 7–6).

Example 7–6 No Resource Viewer (Resource ID as Page View ID)

```
<resource-view authorizerClass="some.service.id.Authorizer3"/>
```

18. Your service should look similar to the one shown in Example 7–7.

Example 7–7 Sample Service Definition for Resource View

```
<service-definitions>
  <service-definition id="mycompany.myproduct.myapp.mycomponent"
                      version="11.1.1.0.0">
  <resource-view taskFlowId="/somepath/somefile.xml#someId"

authorizerClass="mycompany.myproduct.myapp.mycomponent.ResourceAuthorizer"/>
  <resource-type>Custom Object</resource-type>
  <object-types>
    <object-type name="customcomponent123"
```

```
                displayName="Display Name"
                description="Description"/>
  </object-types>

<resource-bundle-class>mycompany.myproduct.myapp.mycomponent.resource.ComponentMes
sageBundle
</resource-bundle-class>
  <name>My Custom Component</name>
  <description>My Custom Component's Description</description>
  <icon>/mycompany/myproduct/myapp/mycomponent/component.png</icon>
  </service-definition>
 ...
</service-definitions>
```

19. Save the `service-definition.xml` file.

7.3.3 Creating a Resource Authorizer Class

The Resource Authorizer interface uses Java interfaces and classes to return useful information about service resources. For example, the `oracle.webcenter.framework.resource.ResourceAuthorizer` interface contains a method that returns the status (available or not available) for a unique resource ID (Example 7–8).

Example 7–8 Method for Returning Status Against a Unique Resource ID

```
/*
 * Check if the passed in resourceIds are viewable by the current user.
 *
 * @param resourceId - the resourceId to check
 * @return - a <code>ResourceInfo</code> with a ResourceStatus
 * of RESOURCE_IS_AVAILABLE if the passed in resourceId
 * is viewable by current user, otherwise returns
 * <code>RESOURCE_IS_NOT_AVAILABLE</code>.
 */
ResourceInfo getResourceInfo(String resourceId, Scope scope);

/*
 * Check if the passed in resourceIds are viewable by the current user.
 *
 * @param resourceIds - the resourceIds to check
 * @return - a List of <code>ResourceInfo</code> with a ResourceStatus
 * of RESOURCE_IS_AVAILABLE if the passed in resourceId
 * is viewable by current user, otherwise returns
 * <code>RESOURCE_IS_NOT_AVAILABLE</code>.
 */
List<ResourceInfo> getResourceInfo(List<String> resourceIds, Scope scope);
```

The `oracle.webcenter.framework.resource.ResourceInfo` interface exposes the following method:

```
public ResourceStatus getResourceStatus();
```

The `oracle.webcenter.framework.resource.ResourceInfo.ResourceStatus` enumeration includes the following possible values:

Example 7–9 Possible ResourceStatus Values

```
/**
```

```
 * Used to describe a related object's status.
 *
 * Certain operations may be performed by relationship service depending
 * on what object status it receives from the service.  For example,
 * if it receives {@link #RESOURCE_DOES_NOT_EXIST} it may remove all
 * relationships associated with this object from the relationship schema
 * so future calls are not wasted on this object.
 */
 public enum ResourceStatus
 {
   /**
    * Used to describe a resource that is available.
    */
   RESOURCE_IS_AVAILABLE,

   /**
    * Used to describe a resource that does not exist.
    */
   RESOURCE_DOES_NOT_EXIST,

   /**
    * Used when the user does not have view access on the resource.
    */
   USER_DOES_NOT_HAVE_ACCESS;
 }
```

The only method in the `oracle.webcenter.framework.resource.ResourceUrlRewriter` interface writes the URL, after a unique identifier is passed in. This URL can be an absolute URL or it can be one relative to the faces context root of the owning application.

Example 7–10 Resource URL Rewriter Method

```
public String rewriteUrl(String id);
```

7.3.4 Specifying the View Handler and Navigation Behavior

You can develop your own resource action view handler -- for example, to cause resources to display in a new tab on the current page -- and specify whether you want to use the ADF inline popup, navigate using the DefaultResourceActionHandler (which opens in the same browser window to display the resource full screen navigation) or use the PopUpResourceActionHandler (which opens a new browser popup window).

Clicking a RAH-enabled link or button in a Framework application invokes the built-in default resource action view handler. This opens the resource in a resource viewer task flow by navigating from the browser to the resource using the same browser window.

You can change this behavior in `adf-config.xml` by specifying either the built-in popup view handler or a custom view handler in the `<resource-handler>` element. Additionally, you can specify whether the default action should invoke the view handler or display as an ADF inline popup in the current window.

Example 7–11 shows the XML added to `adf-config.xml` to declare that `MyAppResourceActionHandler` should be used as the view handler and that in the first instance, the resources should be displayed using an ADF inline popup.

Example 7–11 Specifying a Custom View Handler

```
<wpsC:adf-service-config
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
   <resource-handler
       class="com.mycompany.myapp.MyAppResourceActionHandler"
       display-as-popup="true" />
</wpsC:adf-service-config>
```

where the `wpsC:` namespace is defined as follows:

```
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
```

Along with the `DefaultResourceActionHandler`, the RAH framework also includes the `PopUpResourceActionHandler`, which invokes a new browser popup when navigating, rather than performing a full browser window navigation. This is similar to the behavior of WebCenter Portal applications in previous releases.

Example 7–12 shows how to specify the popup view handler.

Example 7–12 Specifying the PopUp View Handler

```
<wpsC:adf-service-config
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
    <resource-handler
class="oracle.webcenter.framework.resource.view.PopUpResourceActionHandler" />
</wpsC:adf-service-config>
```

Applications can invoke their own resource action view handlers by creating a class implementing the `ResourceActionHandler` interface or by extending the `AbstractResourceActionHandler` base class and specifying the new view handler in the class attribute of `<resource-handler>` in `adf-config.xml`. In the following scenarios, the custom view handler class is invoked when a RAH link is clicked:

- The ADF inline popup is not being used.
- The **Maximize** button is clicked on the ADF inline popup.

> **See Also:** *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal* for the service framework for more information about creating a custom resource action view handler; in particular, the `AbstractResourceActionHandler` class

In the `<resource-handler>` tag, the `display-as-popup` attribute defines the initial behavior for actioning an RAH link. This can be overridden on the RAH link itself in the `resourceActionBehavior` tag using the `useResourcePopup` attribute with the values of `always`, `never` or `appDefined`.

If `useResourcePopup` is not set, or if it is set to `appDefined` on the RAH link, then the `display-as-popup` attribute in `adf-config.xml` is considered. If this is set to `true`, then RAH invokes the ADF inline popup instead of invoking the resource action view handler. This displays a mini-resource-view of the resource in an ADF inline popup within the current page. The user can close or maximize this. If the user clicks the **Maximize** button, then the popup is closed and the resource action view handler is invoked.

By default, the ADF inline popup is used in Framework applications only for specific cases for some task flows where a full navigation does not make sense. For example, search results may include the resource's author. When you click this user name link,

you see the popup summary profile of the author without navigating away from the search result.

These task flows have `useResourcePopup` set to `always` in their `resourceActionBehavior` tags. You can change this by customizing the task flow.

In deployed applications, the `<resource-handler>` settings in `adf-config.xml` can be added and updated using WLST commands. For command syntax and examples, see the sections, "setWebCenterServiceFrameworkConfig" and "getWebCenterServiceFrameworkConfig" in *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*.

7.4 Configuring General Settings for Your Services

Optionally, you can enable users to set the time zone, the date and time format, the language (locale), and the accessibility mode for the services you add to your application. You can provide these capabilities through the WebCenter Portal General Settings service. This service contains values that you can access either directly from the JSF page(s) in your Framework application or indirectly through an API. The General Settings service covers the following areas:

- **Time Zone:** By default, displays the time defined by the server running the application. This is based on `java.util.Timezone`; for example, `GMT+0800`. It also supports the format `Region/Country`; for example, `Europe/Amsterdam`.
- **Date/Time:** By default, configured to the style `java.text.DateFormat.SHORT`. This is based on `java.text.DateFormat`; for example, `java.text.DateFormat.MEDIUM`. Supported values are `SHORT`, `MEDIUM`, `LONG` and `FULL`.
- **Language:** By default, defined by the locale of the user's browser. This is based on `java.util.Locale`; for example, `fr-CA`.
- **Accessibility Settings:** A value to use with the `accessibility-mode` setting in the `trinidad-config.xml` file. Valid values are `default`, `inaccessible`, or `screenReader`.
- **Application Skin:** A value to use with the `skin-family` setting in the `trinidad-config.xml` file. This is based on the Trinidad `skin-family` attribute. Valid values are defined by skins included with the application.

To use the service directly from your application, use Expression Language in your JSF pages to access the `generalSettings` managed bean. The style is based on the `java.util.Timezone`, and the pattern is based on `java.text.SimpleDateFormat`.

When you add `generalSettings` as the value attribute of an ADF component, for example `af:activeOutputText`, you can open the Expression Builder, then navigate to **JSF Managed Beans > generalSettings** to view the preference values for the bean.

The General Settings service supports both pattern-based and style-based formats. If a pattern is not specified, then the style is used. By default, the patterns are not specified or null.

Figure 7–9 shows the Insert EL Expression dialog with General Settings.

Figure 7–9 Expression Builder for the generalSettings JSF Managed Bean

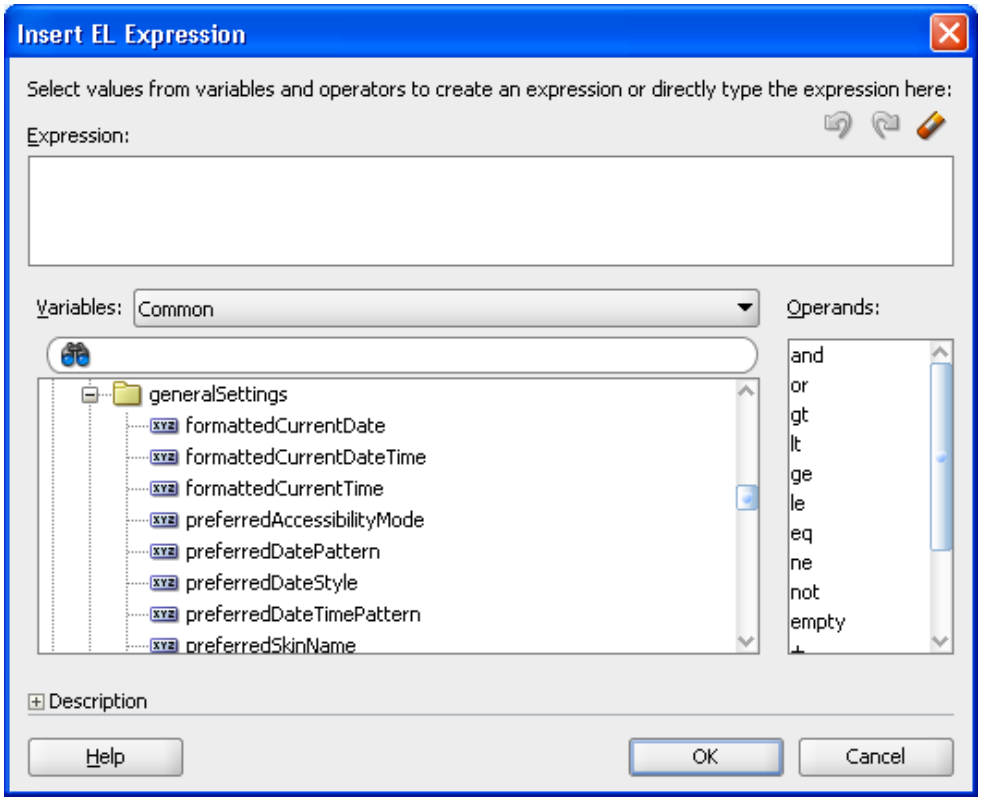


Table 7–4 describes the preference values for the General Settings service.

Table 7–4 General Settings Managed Bean Preference Values and Descriptions

General Settings Preference Value	Description
`formattedCurrentDate`	Displays the current date in the user's selected locale.
`formattedCurrentDateTime`	Displays the current date and time in the user's selected locale.
`formattedCurrentTime`	Displays the current time in the user's selected locale.
`preferredAccessibilityMode`	Preferred accessibility mode (`default`, `inaccessible`, or `screenReader`) for use in the `accessibility-mode` setting in `trinidad-config.xml`.
`preferredDateStyle`	Java date style to be used in `dateStyle` attributes when displaying dates and times.
`preferredTimeStyle`	Java time style to be used in `timeStyle` attributes when displaying dates and times.
`preferredDatePattern`	Java date pattern to be used when displaying dates, and not times.The pattern must be a valid `java.text.SimpleDateFormat` pattern; for example: `dd-MMM-yyyy`. This takes precedence over `preferredDateStyle`.
`preferredTimePattern`	Java time pattern to be used when displaying times, and not dates. This takes precedence over `preferredTimeStyle`.

Table 7–4 (Cont.) General Settings Managed Bean Preference Values and Descriptions

General Settings Preference Value	Description
`preferredDateTimePattern`	Java date/time pattern to be used when displaying dates and times. This takes precedence over `preferredDateStyle` and `preferredTimeStyle`.
`userTimeZone`	Java time zone to be used in `timeZone` attributes when displaying dates and times.
`preferredSkinName`	Preferred skin name to use with the `skin-family` setting in `trinidad-config.xml`.

For example, to display the current date and time in the user's selected locale, time zone, and format, add the following to your page:

Example 7–13 Code for Displaying Current Date and Time in User's Locale

```
<af:outputText value="#{generalSettings.formattedCurrentDateTime}"/>
```

Or, to display a specific date and time:

Example 7–14 Code for Displaying Specified Date and Time

```
<af:outputText value="#{row.dateTimeValue}">
    <af:convertDateTime type="both"
                        dateStyle="#{generalSettings.preferredDateStyle}"
                        timeStyle="#{generalSettings.preferredTimeStyle}"
                        timeZone="#{generalSettings.userTimeZone}"
                        locale="#{facesContext.externalContext.requestLocale}" />
                        pattern="#{generalSettings.preferredDateTimePattern}"
    </af:outputText>>
```

To take advantage of the preferred accessibility mode, you must add an EL (Expression Language) expression to the application's `trinidad-config.xml` file to set the accessibility mode; for example:

Example 7–15 Setting the Accessibility Mode in trinidad-config.xml

```
<accessibility-mode>#{generalSettings.preferredAccessibilityMode}</accessibility-m
ode>
```

7.4.1 Building a Preferences User Interface

You can build a user interface using the General Settings API to enable users to either accept the default settings or apply their desired settings. The API contains the same preference values described in Table 7–4.

Any service in WebCenter Portal that displays a date and time uses the settings configured in the General Settings service. For information on using the General Settings Service API to build a preferences user interface, see the Javadoc for `oracle.webcenter.generalsettings`. Specifically, the `oracle.webcenter.generalsettings.model` package contains APIs to get and set preference values for each of the settings. Use these APIs to build a custom user interface to allow users to view and set their preferred values.

> **See Also:** *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*

7.4.2 Using General Settings to Specify the User's Locale

By default, ADF obtains the user's locale from the user's browser settings. You can change this behavior to have the user's preferred locale be set in the application using a Preferences UI.

To make the behavior change, include `oracle.webcenter.generalsettings.model.filter.GeneralSettingsLocaleFilter` as a servlet filter in `web.xml` and `adfBindings`. This servlet filter sets the default locale (ADF Locale) from the General Settings service Locale preference instead of from the user's browser.

Add the General Setting locale filter to `web.xml` with the following rules:

Define the filter. For example:

```
<filter>
   <description>
      WebCenter General Settings locale setting.
   </description>
   <filter-name>
      generalSettingsLocaleFilter
   </filter-name>
   <filter-class>

oracle.webcenter.generalsettings.model.filter.GeneralSettingsLocaleFilter
   </filter-class>
</filter>
```

Add the filter after the `adfBindings` filter with the same servlet-name. For example:

```
<filter-mapping>
   <filter-name>
      generalSettingsLocaleFilter
   </filter-name>
   <servlet-name>Faces Servlet</servlet-name>
</filter-mapping>
```

If this filter *is not* set, then the locale is sourced first from the client browser locale setting, and then from the ADF locale setting.

If this filter *is* set, then the locale is sourced from the General Settings service. If the General Settings locale preference does not exist, then it reverts to the non-filter default behavior.

8

Introducing the AviTrust Portal Framework Sample Application

As a companion to this guide, a sample enterprise application was developed using Oracle Fusion Middleware 11g Release 1 (11.1.1.5), which includes Oracle WebCenter Portal: Framework and Oracle WebCenter Portal: Spaces. The application, called the AviTrust Portal Demonstration for WebCenter Portal, is referenced in various chapters throughout this guide to illustrate points and provide typical use cases for Oracle developers. It is intended to serve as a powerful learning tool for understanding the new portal features and services available in the latest release of WebCenter Portal.

In this chapter, you'll learn about these features and services, as well as take a quick, high-level tour of the components that were built using WebCenter Portal: Framework.

This chapter includes the following sections:

- Section 8.1, "About the AviTrust Portal Framework Application for WebCenter"
- Section 8.2, "Downloading the AviTrust Portal Framework Application"
- Section 8.3, "Browsing Portal Framework Features at Runtime"
- Section 8.4, "Accessing and Using the Resource Manager"
- Section 8.5, "In-Context Editing of Web Content at Runtime"
- Section 8.6, "Exposing Social Computing Services"
- Section 8.7, "WebCenter Framework Components in the AviTrust Portal Demo"

8.1 About the AviTrust Portal Framework Application for WebCenter

The AviTrust Portal Demo for WebCenter is a robust, rich-media portal application built for the enterprise, which demonstrates some of the advanced features available in Oracle Fusion Middleware, as well as WebCenter Portal: Framework. The application is based on a hypothetical online banking and financial services website -- the AviTrust Listening Bank.

Using this public-facing, extranet portal (shown in Figure 8–2), customers can browse and navigate various banking services offered by the AviTrust Bank, such as opening checking and savings accounts or applying for loans, through the bank website. Customers can also take advantage of a wide range of banking and financial services tailored for both small businesses and larger commercial enterprises, like setting up custom online accounts for access to the portal.

In this portal application, WebCenter Portal and WebCenter Portal: Framework are used to provide a host of Web 2.0 capabilities that include content-driven navigation, personalization scenarios, and integration of new WebCenter Portal: Services,

including polls, blogs, RSS, discussion forums, tags, links, social networking and activity streams. These services are directly embedded into the AviTrust sample portal application.

By design, the AviTrust banking portal application was created (hypothetically) to reduce transaction costs and offer a wide range of convenient banking services to its online customers.

For example, the AviTrust banking portal allows authenticated customers to:

- Log in securely to browse various banking services and online offers
- Open checking and savings accounts
- Apply for loans online and take advantage of other banking services available for small businesses and large enterprises
- Participate in online forum discussions, wikis, polls, blogs and other social computing activities

8.2 Downloading the AviTrust Portal Framework Application

To view and run the demo application, you need to install Oracle JDeveloper 11*g* (11.1.1.4) and the Oracle WebCenter Portal extension (11.1.1), as described in Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper." You then need to download the application for this demonstration. Once the application is installed and running, you can view the application at design time in Oracle JDeveloper and at runtime by logging in as a user or administrator.

The AviTrust Portal Demo source distribution is available via the WebCenter Oracle Technology Network (OTN) page for Demos and Samples.

Sample files to import into your WebCenter Portal application, which include both project and archive files, available for download as a ZIP (or tgz) archive, will be made available here.

`http://www.oracle.com/technetwork/middleware/webcenter/portal/documentation/avitrust-522536.html`

The ZIP archive also includes instructions on how to load and install the project content.

Sample code for developers and administrators is available at `http://www.oracle.com/technetwork/indexes/samplecode/index.html`

Release downloads for Oracle WebCenter Portal Developers are available at

`http://www.oracle.com/technetwork/middleware/webcenter/portal/downloads/index.html`

8.3 Browsing Portal Framework Features at Runtime

Once you have downloaded and installed the AviTrust sample portal application, you can run the application and navigate to the components that were built using WebCenter Portal. This section will take you on a visual tour of the application and show you how the components were built using WebCenter: Framework.

The section focuses on how the various Oracle WebCenter Portal components display at runtime. You can examine the application at design time in Oracle JDeveloper. For more information on how you can add a service or feature to your application, refer to the appropriate chapter in this guide.

In broad outline, some of the new features illustrated by the AviTrust sample application in this release of WebCenter Portal include:

- Improved portal navigation, with pages based on multiple levels of navigation and content based on different page templates. For information about how to expose the navigation model in your portal, see Chapter 14, "Visualizing Your Portal Navigation."
- Publication of HTML content from Oracle WebCenter Content by means of Oracle Content Presenter. For more information, see Chapter 29, "Creating Content Presenter Display Templates."
- Content integration with in-context editing of HTML documents stored on Oracle Content Server, as well as in-context editing of Site Studio definition files. Use of content-driven navigational links For information on how to create the navigation model for your portal, refer to Chapter 11, "Building a Navigation Model for Your Portal."
- Personalization, which enables developers to render content from the Content Presenter, based on a personalization scenario. For more information, refer to Chapter 67, "Personalizing WebCenter Portal Applications."
- Resource management, enabling developers to manage portal resources, like pages, templates, catalogs, and skins, at runtime. For more information, see Chapter 13, "Adding Resources to Your Portal."
- Integration of WebCenter services, like polls, forums, blogs and wikis, for enhanced user experience in social computing. For more information, see Chapter 37, "Integrating the Polls Service."

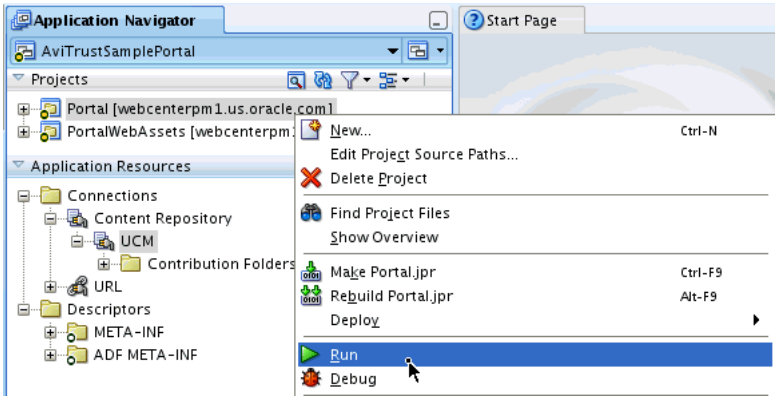
8.3.1 Exploring the AviTrust Sample Portal Home Page

To understand how the components of the banking portal were constructed, you can begin by launching and deploying the application in JDeveloper and then taking a look at the AviTrust Sample Portal Home page, as displayed in a web browser. The Home page will serve to illustrate some of the powerful, new features available in WebCenter Portal.

To view these features:

1. Open the AviTrust Sample Portal application in Oracle JDeveloper.
2. In Application Navigator, select the `Portal [webcenterpm1.us.oracle.com]` project and right-click the **Run** icon, as shown in Figure 8–1.

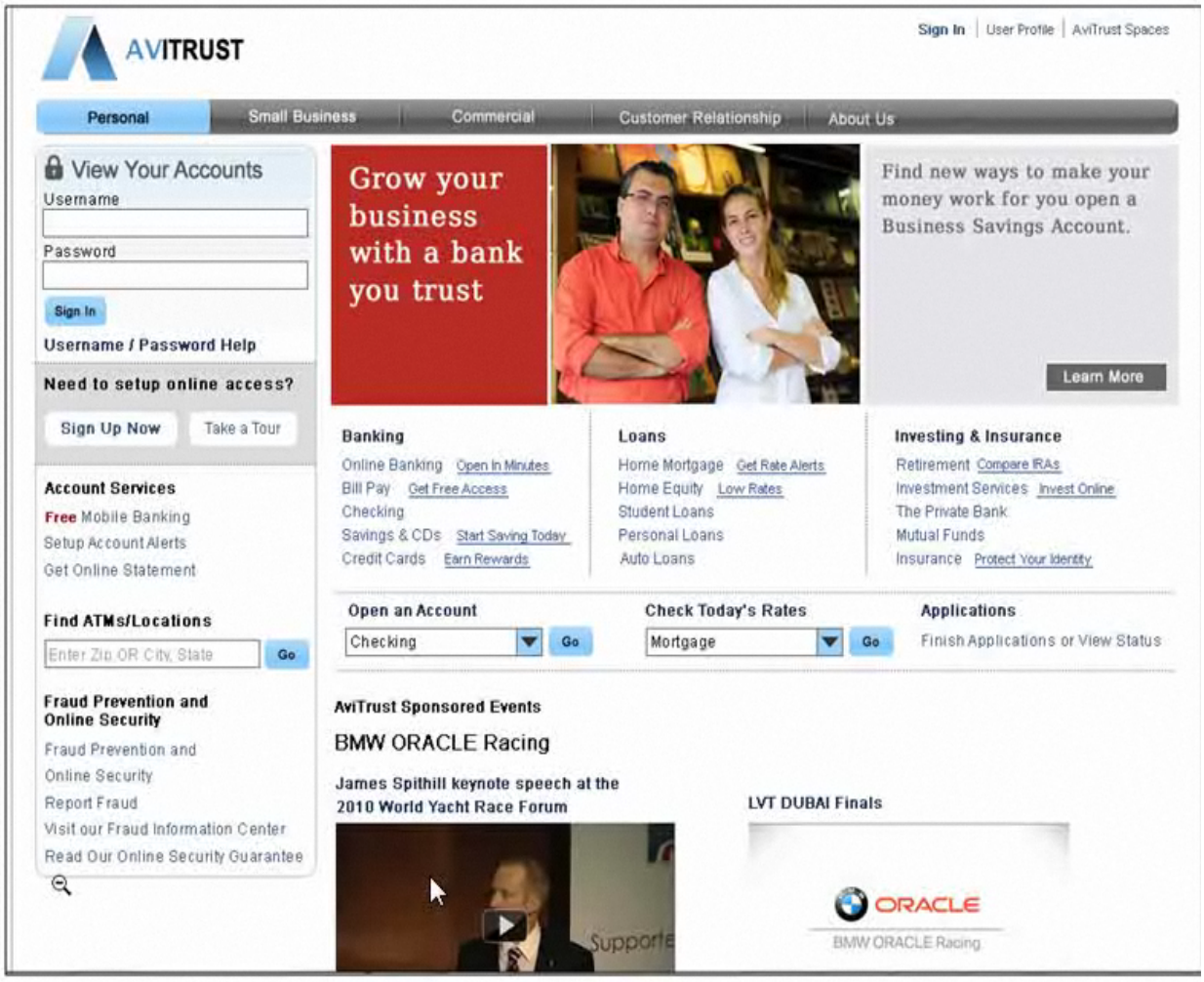
Figure 8–1 The AviTrust Sample Portal Application Launched in Oracle JDeveloper



If the Integrated WebLogic Server is not running, this action starts the server and (re)deploys the application. By default, the file `index.html` (or another page that it redirects to) displays in the browser.

The Home page of the sample portal application now appears in a web browser, as shown in Figure 8–2.

Figure 8–2 The AviTrust Home Page Displayed in a Web Browser



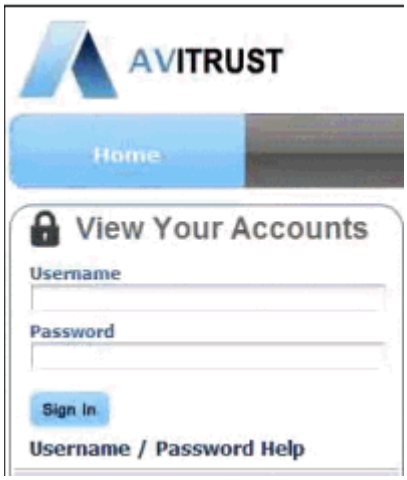
****

On the portal Home page, secure user login is provided for authenticated bank customers by two points of entry:

- Either by clicking the Sign-In link in the upper right portion of the portal
- Or by entering their User Name and Password in the designated fields in the left portion of the Home page (shown in Figure 8–3).

In addition, administrators can set up security for authenticated users and provide delegated administration, while users manage their account and banking transactions. For more information, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."

Figure 8–3 Secure Customer Login to Accounts in the AviTrust Home Page



8.3.2 Navigating the Site Using Tabs

Tabbed navigation provides banking customers with the most flexible and intuitive way of exploring the site.

The navigation model, rendered as a fly out menu on the template header, is based on an HTML template and a Cascading Style Sheet (CSS) file created by the AviTrust graphics design team. With four tabs embedded in the navigational bar (shown in Figure 8–4), customers can easily browse and access other pages and levels of information and services available on the banking website.

Figure 8–4 The Navigation Bar on the Home Portal Page with Four Tabs



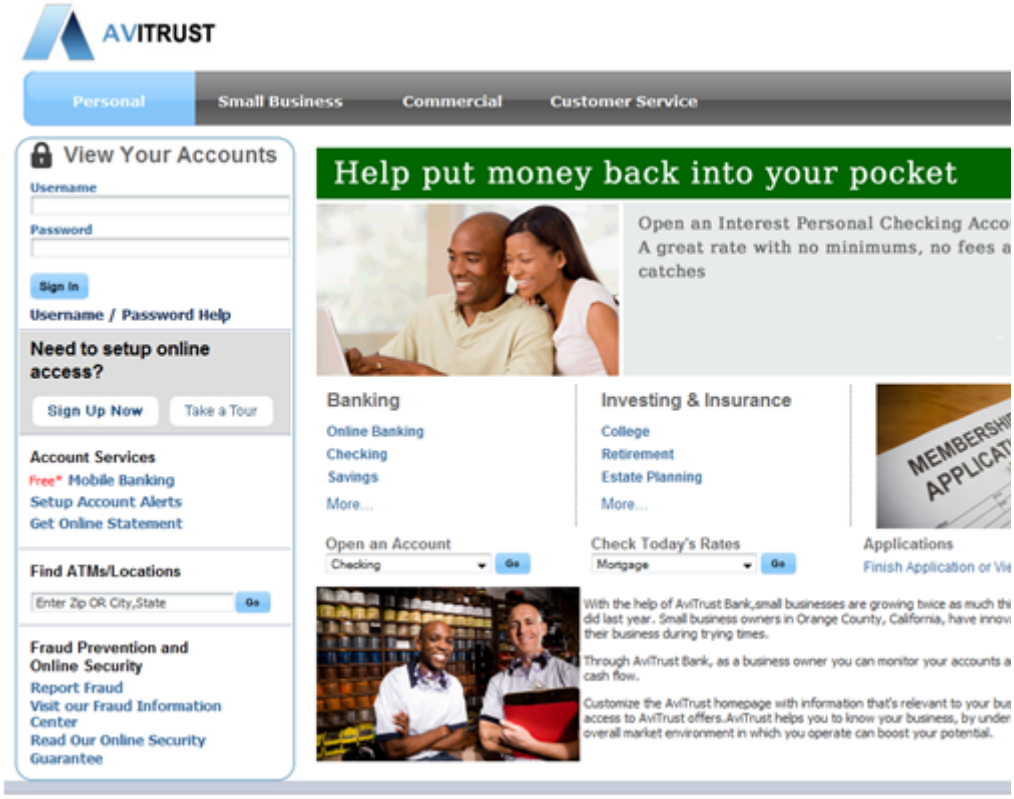
To navigate the site using tabs:

1. Move your cursor over the **Personal** tab in the navigation bar at the top of the page. A group of banking services are displayed, with links that customers can then click to access pages specific to their personal banking needs in the portal, as shown in Figure 8–5.

   Note that as you move your cursor over this and the other tabs in the navigation bar (illustrated subsequently in Figure 8–6, Figure 8–7, and Figure 8–8), the AviTrust Portal Application uses a different site template. There is a notable switch from the horizontal top-level, tab-style navigation to a hybrid model. This model uses horizontal tabs on the top and vertical links for side navigation. The WebCenter navigation components here are designed to be flexible yet easy to manage for purposes of branding partner and sub sites.

   For information on how to create the navigation model for your portal, refer to Chapter 11, "Building a Navigation Model for Your Portal."

Figure 8–5 The Personal Tab Selected in the Navigation Bar with Links to Other Pages and Services



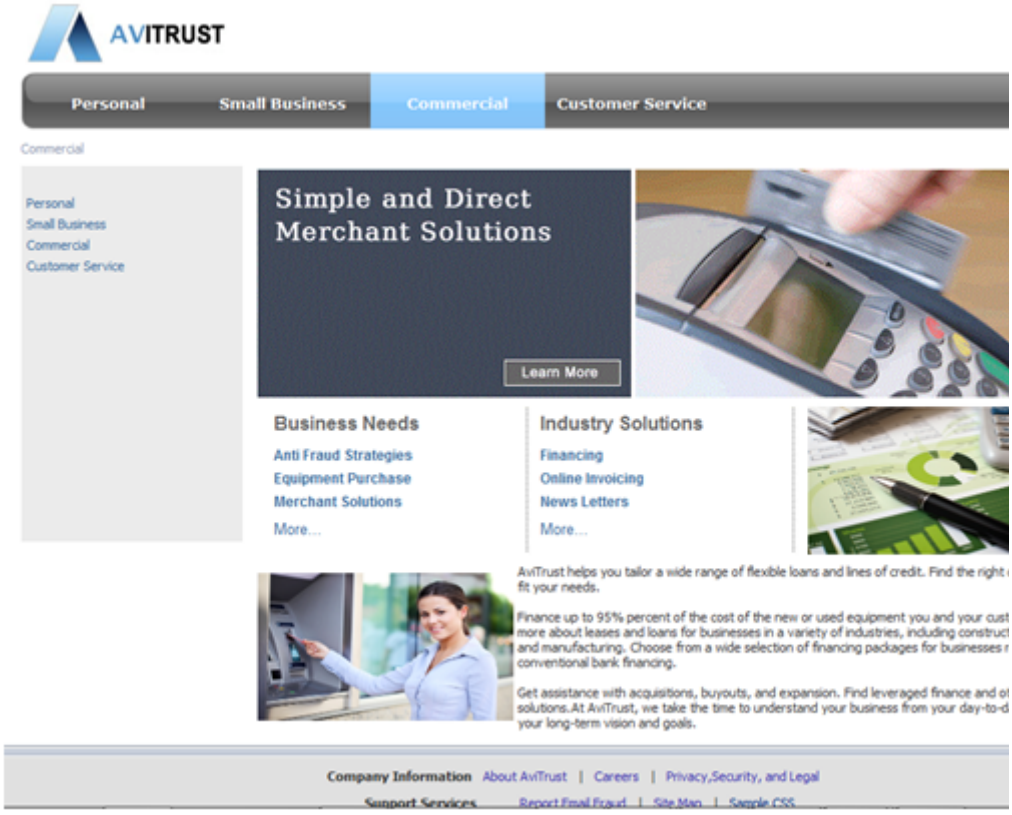
2. Now moving your cursor over the **Small Business** tab in the navigation bar at the top of the page, you find another group of banking services displayed, this time with links that customers can click to access pages specific to the needs of small businesses in the portal, as shown in Figure 8–6.

Figure 8–6 The Small Business Tab Selected in the Navigation Bar with Links to Other Pages and Services



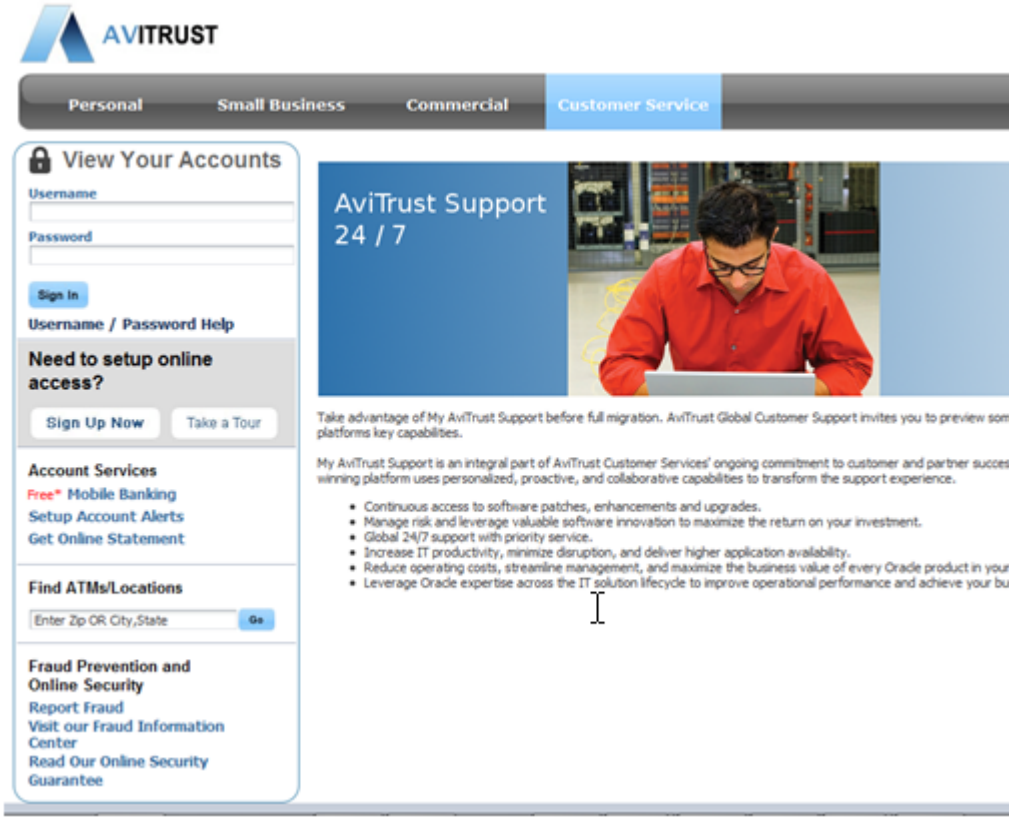
3. If you move your cursor over the **Commercial** tab in the navigation bar at the top of the page, your web browser will display another group of banking services, with links that commercial customers can click to access pages specific to commercial banking needs in the portal, as shown in Figure 8–7.

Figure 8–7 The Commercial Tab Selected in the Navigation Bar with Links to Other Pages and Services



4. By moving your cursor over the **Customer Support** tab in the navigation bar at the top of the page, your web browser will display yet another group of banking services, with links that users can click to access customer support pages in the portal, as shown in Figure 8–8.

Figure 8–8 The Customer Support Tab Selected with HTML Contents Displayed

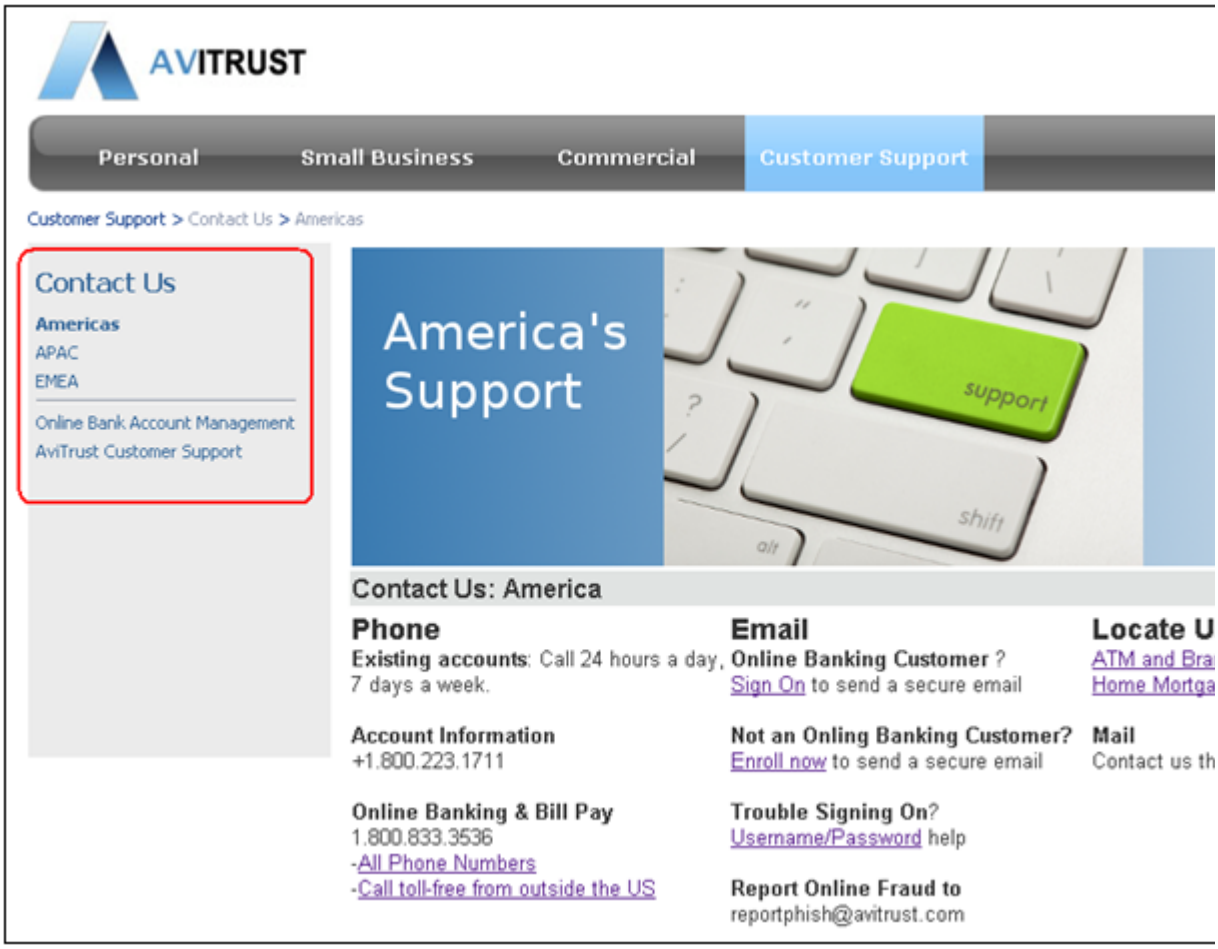


In each case, customers browsing and accessing the AviTrust portal will find ease of access in navigating to pages for information that are relevant to their various banking needs.

5. Click the **Americas** link on the **Contact Us** page, as shown in Figure 8–9. Note that the pages under Contact Us, like Americas, APAC and EMEA, are based on a different template, which provides customer navigational links on the left side of the page.

   The bread crumb on the top of the page lets you identify your current location within the site, and also provides users with easy access back to the top-level pages.

Figure 8–9 The America's Support Page Following the Bread Crumb from the Contact Us Link



8.4 Accessing and Using the Resource Manager

The AviTrust Portal application demonstrates how users or administrators with appropriate privileges can easily manage portal resources at runtime.

At runtime, the Resource Manager provides access to administration tools for the resources available to the portal. For example, using the Resource Manager, the site administrator can perform the following tasks:

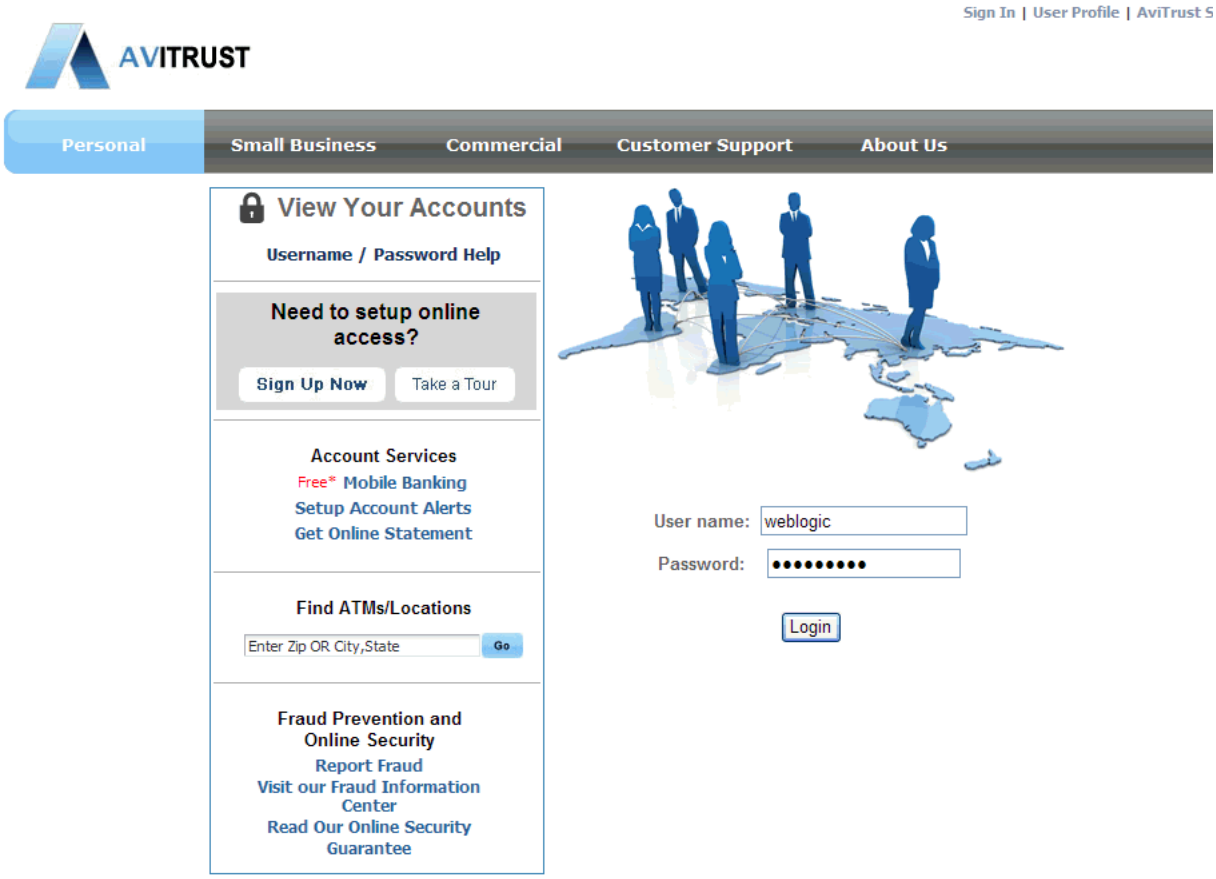
- Create new resources, like navigation models, page templates and mashup styles
- Edit existing resources
- Upload resources exported from other applications, among other tasks

For more information, see Chapter 13, "Adding Resources to Your Portal."

To access and use the Resource Manager at runtime:

1. Click the Sign-In link in the upper right corner of the Home page. A page appears with fields to enter User Name and Password, shown in Figure 8–10.

Figure 8–10 The Login Page Displayed in a Web Browser



2. Log in with the following credentials:

   Username: `weblogic`

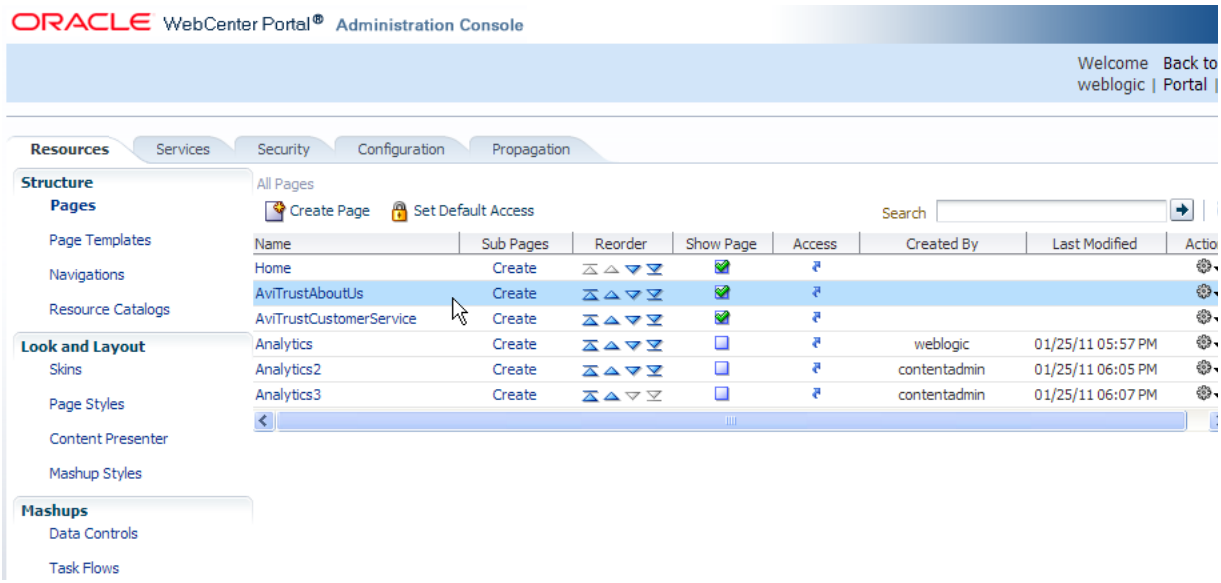
   Password: `welcome1`

   The Administration Console now appears in the web browser. As an administrator, you can make changes at runtime and manage portal resources accordingly.

3. In the Administration Console, click the Resources tab and select the Pages node in the Structure menu, shown in Figure 8–11. You can now create sub pages, specify the order and display of those pages, as well as perform editing tasks on those pages. <

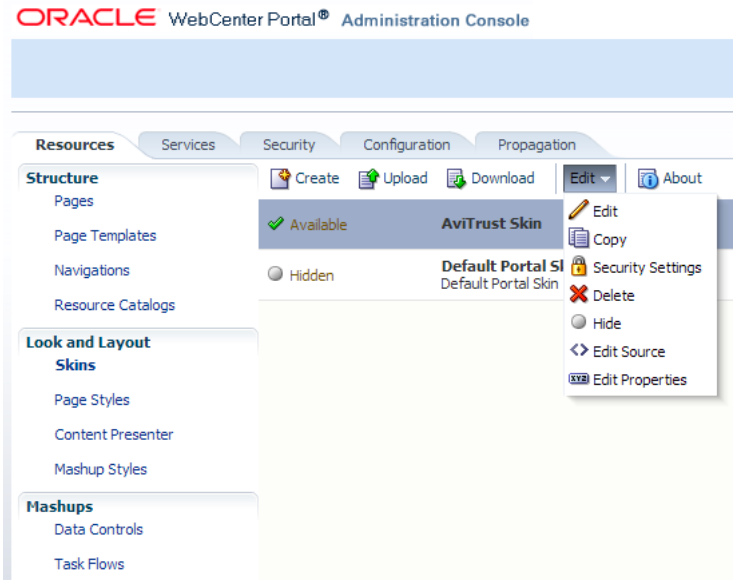
   For more information, see Section 9.11.2, "Runtime WebCenter Portal Administration Console."

Figure 8–11 The Administration Console with Pages Selected



4. In the Administration Console, click the Resources tab and select the Skins node in the Look and Layout menu, shown in Figure 8–12. You can now edit, change or modify the skins used in the portal.

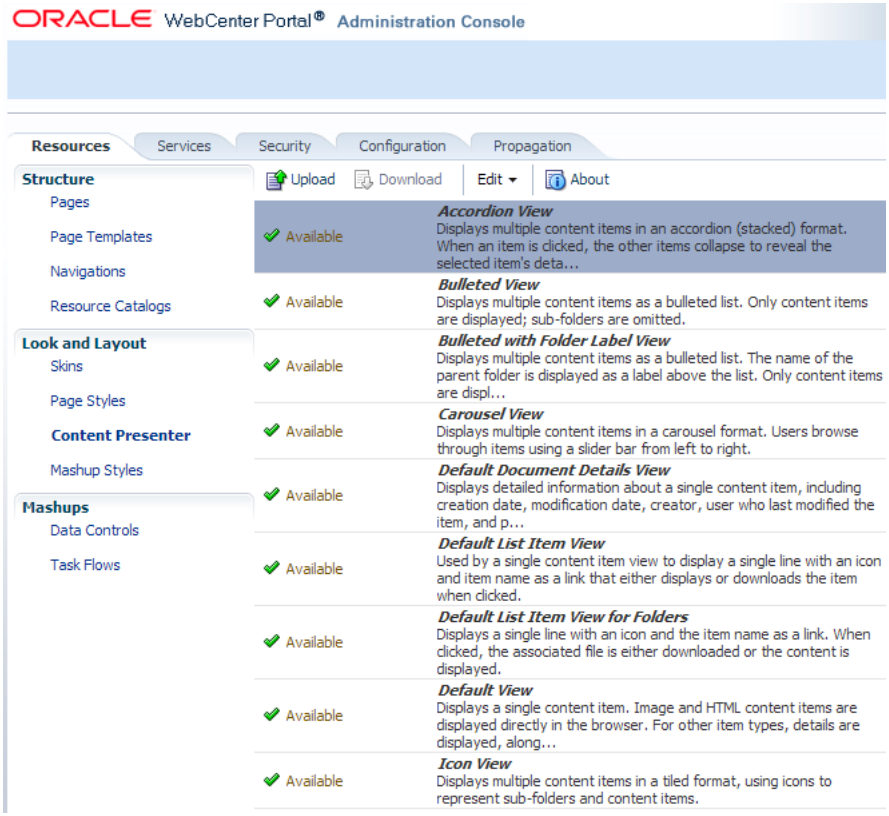
Figure 8–12 The AviTrust Skin Selected in the Look and Layout Menu



For more information, see Chapter 15, "Creating and Managing Skins."

5. In the Administration Console, click the Resources tab and select the Content Presenter node in the Look and Layout menu, shown in Figure 8–13. You can now change the look and layout of your portal, using the Content Presenter to upload or edit item, folder or labels views.

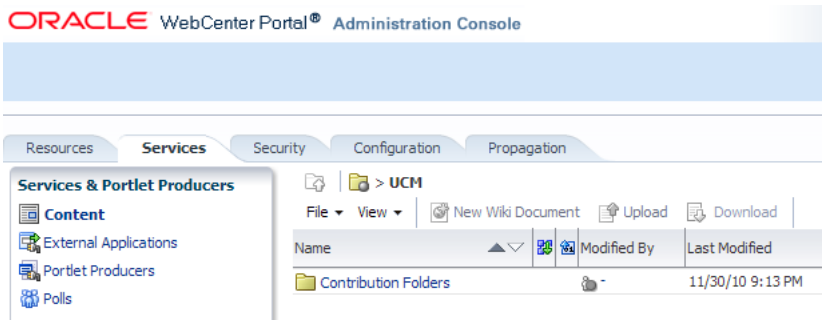
Figure 8–13 The Content Presenter Node Selected in the Look and Layout Menu



For more information, see Chapter 29, "Creating Content Presenter Display Templates."

6. In the Administration Console, click the Services tab and select the Content node in the Services & Portlet Producers menu, shown in Figure 8–14.

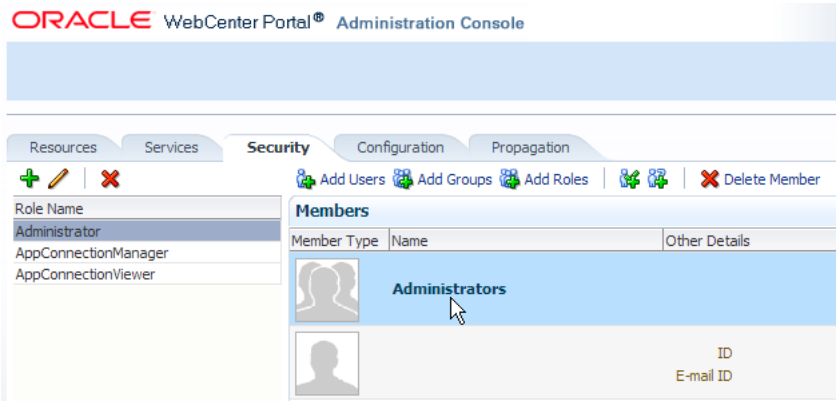
Figure 8–14 The Content Node Selected in the Services & Portlet Producers Menu



The Services tab with Contribution folder selected enables you to access content stored in Oracle WebCenter Content on Oracle Server.

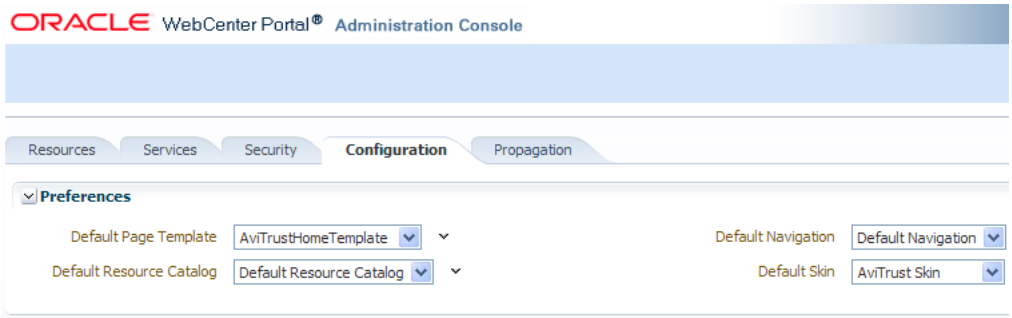
7. In the Administration Console, click the Security tab and select the Administrator node in the Role Name menu, shown in Figure 8–15. In so doing, you can set security access and privileges for the portal Administrator.

Figure 8–15 The Security Tab Selected in the Administrator Node



8. In the Administration Console, click the Configuration tab and select the Default Page Template node in the Preferences menu, shown in Figure 8–16. In so doing, you can change the default page template to another template of your choice.

Figure 8–16 The Configuration Tab Selected in the Preferences Menu



8.5 In-Context Editing of Web Content at Runtime

All portals need to enable content authors to make changes quickly and easily at runtime. Using WebCenter Portal's direct integration with Oracle Enterprise Content Management (ECM), authors can create and edit HTML documents with a minimum of effort.

For example, you can retrieve a template entry for a press release and then ensure that your new press release follows your corporate standard, thus easily evolving the site at runtime and still professionally managing the source content for your entire site.

To accomplish this, you log in to the AviTrust Portal as a content author who has privileges to edit content in a web browser, following these steps:

1. Log in with the following credentials:

   Username: `contentadmin`

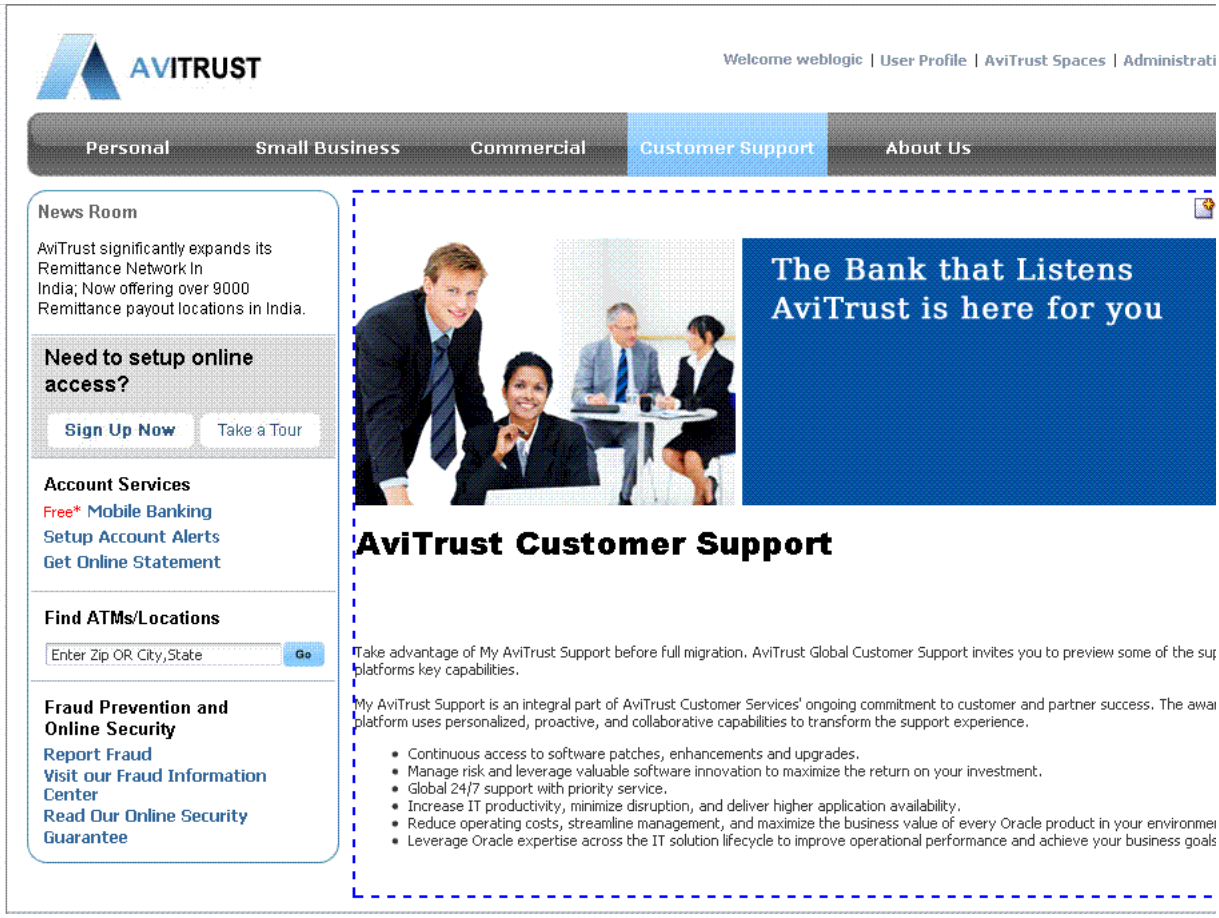
   Password: `welcome1`

2. Click the **Customer Support** tab.

3. Navigate to the center of the portal and press `CTRL+SHIFT+C` on your keyboard.

This takes you into content contribution mode for runtime editing. A dotted line appears around the content to be edited, shown in Figure 8–17.

Figure 8–17 In-Context Editing Enabled on the AviTrust Customer Support Page



4. Navigate to the upper right corner in the dotted area of the page and click the pencil icon on the top right hand side corner, as shown in Figure 8–18. This launches Site Studio Contributor, which enables you to edit the selected web content.

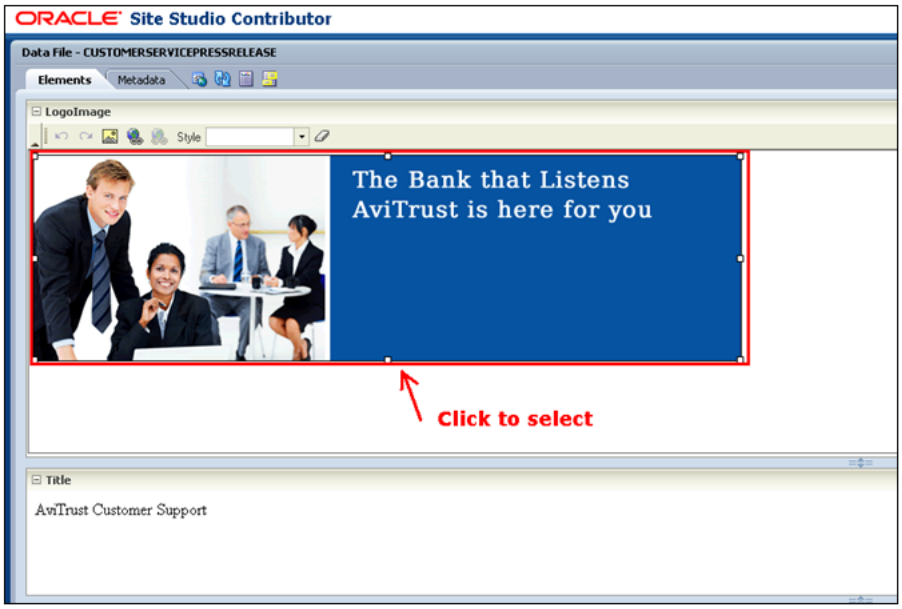
Figure 8–18 The Edit Pencil Selected to Enable Runtime Editing of Web Content



5. In Oracle Site Studio Contributor, click to select the banner image and then delete it, shown in Figure 8–19.

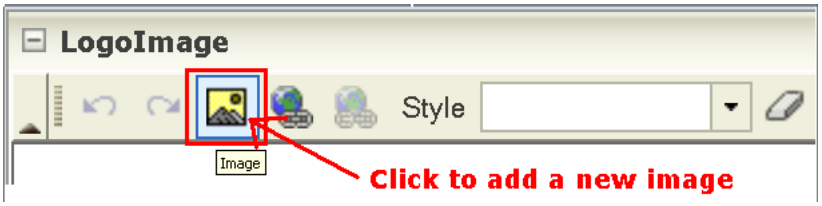
   Note that you can easily replace the images or edit text through the Site Studio Contributor.

Figure 8–19 Oracle Site Studio Contributor With the Content to be Edited



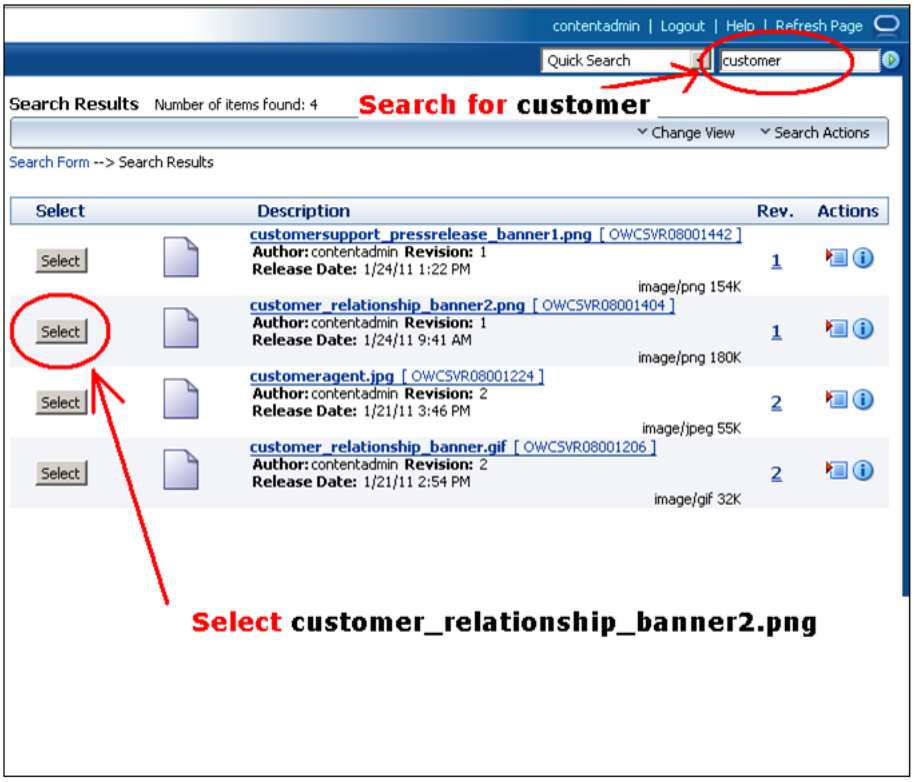
6. Click the image icon (shown in Figure 8–20) to add a new image from the ECM repository.

Figure 8–20 The Add Image Icon Selected



7. Open the Search form and in the search field in the upper right portion of the form, enter the keyword `customer` in the ECM console, as shown in the Figure 8–21, and select the image `customer_relationship_manager2.png`. You will now be able to search any new images uploaded by the content management team and choose an image from the list.

Figure 8–21 Searching for the Keyword Customer in the ECM Console



8. Once the image is replaced, click Save and Close in Oracle Site Studio Contributor, as shown in Figure 8–22.

Figure 8–22 The Save and Close Button Selected in Oracle Site Studio Contributor



9. Click the refresh icon (shown in Figure 8–23).

   You can now refresh the content to see the new image.

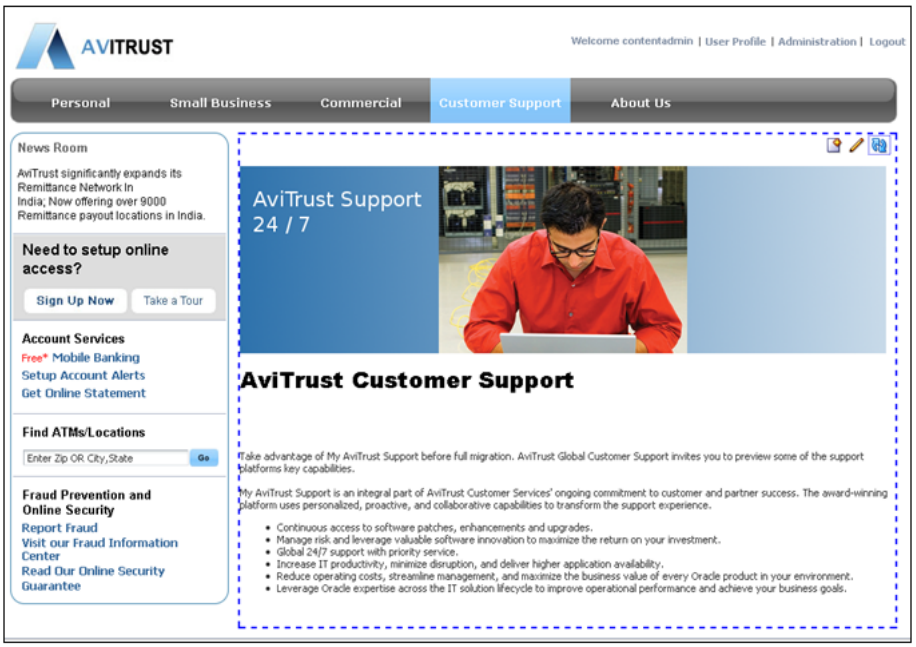
Figure 8–23 The Refresh Icon Selected



10. Press `CTRL+SHIFT+C` again on your keyboard.

    Since you have finished in-context editing of the content stored in ECM, you can come out of contribution mode. The newly edited banner image and content now appears on the AviTrust Customer Support page, as shown in Figure 8–24.

Figure 8–24 The Newly Edited Banner Image and Content on the Customer Support Page



8.6 Exposing Social Computing Services

WebCenter Portal exposes a comprehensive set of social computing services. These include ready-to-use task flows and portlets, enabling developers to become more productive. These services are designed to work together, so that you can add polls, blogs, threaded discussions, wikis and other social computing services to your site with a minimum of effort at runtime.

The **Polls service** lets you create, edit, and take online polls on your application pages. Polls let you survey your audience (such as their opinions and their experience level), check whether they can recall important information, and gather feedback on the efficacy of presentations.For more information, see Chapter 37, "Integrating the Polls Service."

In one scenario, for example, let's suppose an executive at the AviTrust Listening bank wants to poll members of the Online Small Business Community. The executive asks

the website administrator to create a new Poll at runtime on the Polls Administration tab. The website administrator creates the poll and puts it on the home page of the community. A customer then takes the poll and the bank executive views the results. In this case, the poll is implemented at runtime, without having to make changes in JDeveloper at design time.

In another scenario, a small business customer may want to host an event for other customers. In the AviTrust banking portal, the customer creates an Event wiki page, using a specific template to define when and where the event is supposed to take place. Another customer can edit the wiki page by adding a requested agenda item or other information about the event.

The **Wiki service** provides a means for customers to work simultaneously on documents and share ideas, either in the context of communities or as individuals. Customers can use wiki syntax directly, or edit wiki pages as you would any text in a word processor. All changes and versions are tracked within Oracle ECM. Thus, wiki content becomes a true enterprise content resource.

For more information, refer to Chapter 32, "Integrating Wikis and Blogs."

8.6.1 Adding a Discussion Forum

The AviTrust sample application illustrates how easily you can add a **Discussion Forum**, similar to adding other social computing services, like polls, blogs and wikis, to your portal at runtime.

For more information, refer to Section 25.3.2, "Examples: Customizing Task Flows for Custom Framework Applications."

To create, edit and manage the content for a Small Business Forum in the AviTrust banking portal:

1. Log in with the following credentials:

   Username: `contentadmin`

   Password: `welcome1`

2. Click the Small Business tab.
3. Navigate to the Forums link (shown in Figure 8–25) and click it.

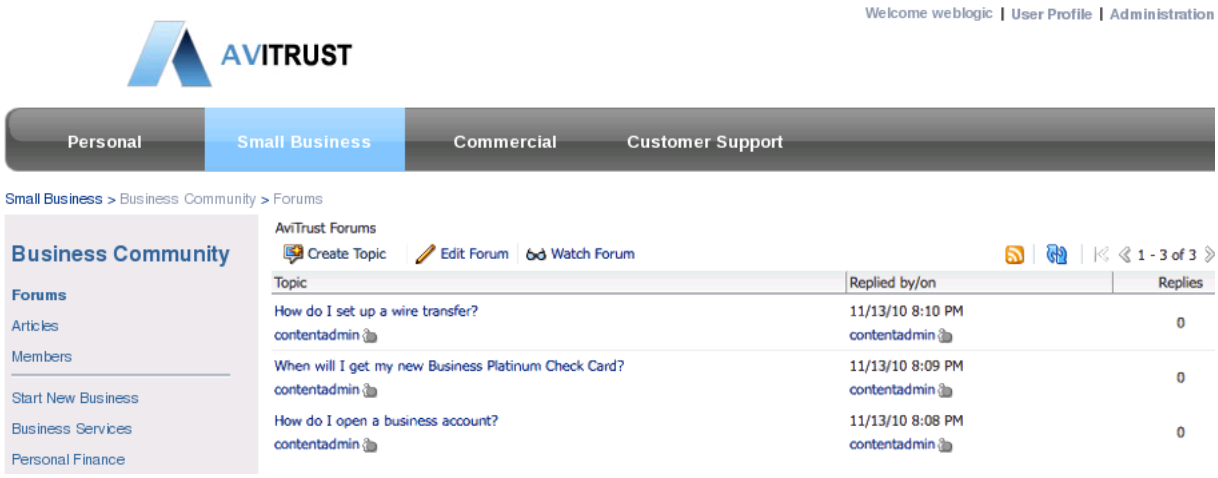
Figure 8–25 The Forums Link in the Small Business Page



4. Navigate from the Small Business link to the Business Community page and click Forums, as shown in Figure 8–26. A list of AviTrust discussion topics and frequently-asked forum questions now appears

   As the content administrator, you can create a topic for the forum, edit its contents and observe the customer activity, such as replies and exchanges, on the forum itself.

Figure 8–26 The Forum Page to Create a Topic and Edit the Content of a Forum Discussion



8.7 WebCenter Framework Components in the AviTrust Portal Demo

Now that you've taken a brief tour of the AviTrust Portal Demo for WebCenter, you can check out these features in depth and use them yourself by referring to the relevant chapters in this guide.

Table 8–1 describes at a high level the Oracle WebCenter Portal features that are included in the AviTrust Portal Demo for WebCenter.

To learn more about the different features, refer to the relevant chapter.

Table 8–1 WebCenter Portal Framework Features in the AviTrust Portal Demo for WebCenter

Feature	Location in the Demo	Chapter
Portal Navigation	Users can navigate the portal using tabs in the navigation bar. Developers can add content-driven navigational links to the portal, like pages, portlets, task flows, files, external applications or web pages in the navigation model.	Chapter 14, "Visualizing Your Portal Navigation."
Content Integration	Authenticated users can edit in-context HTML documents stored on Oracle Content Server, as well as in-context editing of Site Studio definition files.	Chapter 11, "Building a Navigation Model for Your Portal.">
Personalization	Administrators can render content based on a personalization scenario, using the Content Presenter. Content can be based on a query result derived from the a scenario deployed in the Personalization engine.	Chapter 67, "Personalizing WebCenter Portal Applications."

Table 8–1 (Cont.) WebCenter Portal Framework Features in the AviTrust Portal Demo for WebCenter

Feature	Location in the Demo	Chapter
Resource Management	Portal resources, like pages, catalogs, and skins, can be managed at runtime, using Administration capabilities.	Chapter 13, "Adding Resources to Your Portal.""
Integration of WebCenter Services	Administrators can take advantage of social computing services, like polls, blogs, forums and wikis.	Chapter 32, "Integrating Wikis and Blogs."

Part II

Setting Up Your Portal Structure

Part II contains the following chapters:

- Chapter 9, "Understanding Framework Applications"
- Chapter 10, "Understanding the WebCenter Portal Life Cycle"
- Chapter 11, "Building a Navigation Model for Your Portal"
- Chapter 12, "Designing the Look and Feel of Your Portal"
- Chapter 13, "Adding Resources to Your Portal"
- Chapter 14, "Visualizing Your Portal Navigation"
- Section 15, "Creating and Managing Skins"
- Chapter 16, "Creating and Managing Resource Catalogs"
- Chapter 17, "Adding and Using the Resource Manager"

9

Understanding Framework Applications

This chapter introduces the architecture, components, and features of a WebCenter Portal: Framework application. This chapter includes these topics:

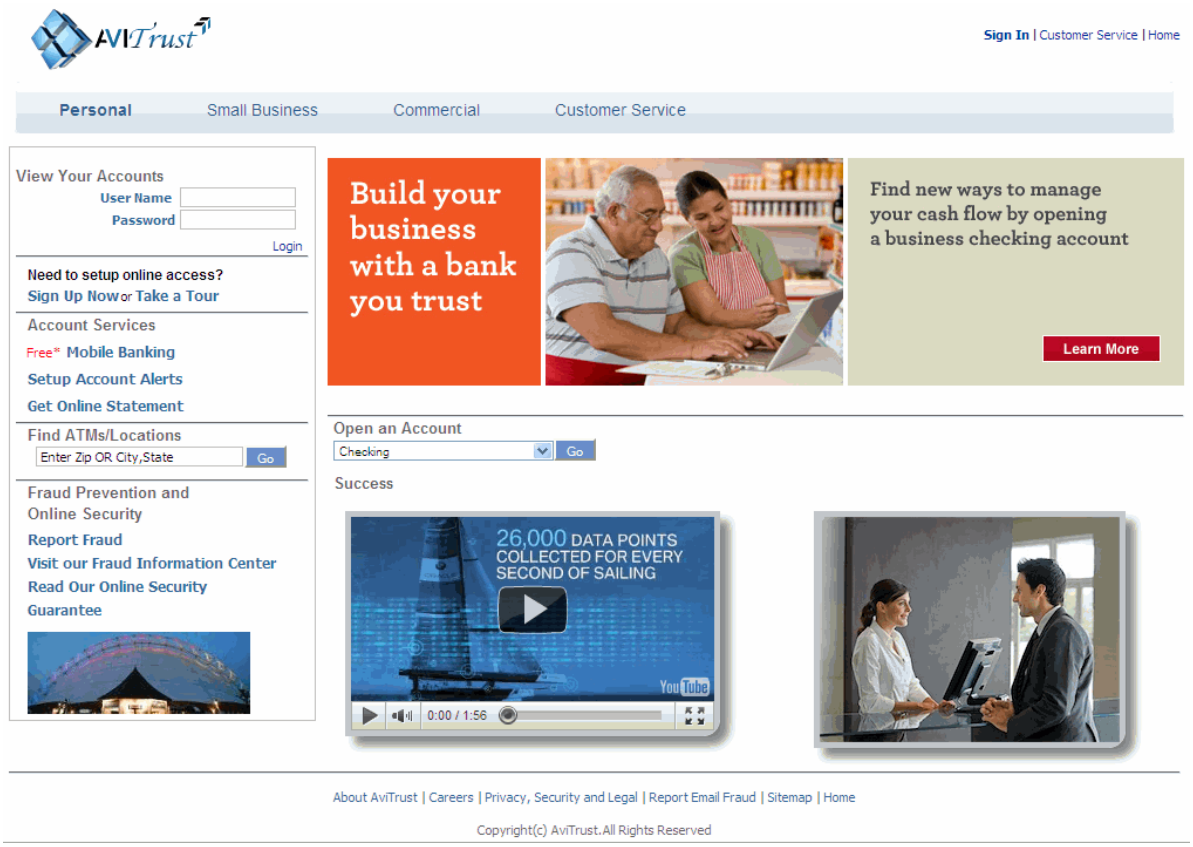
- Section 9.1, "What is a Framework Application?"
- Section 9.2, "What is Oracle WebCenter Portal: Framework?"
- Section 9.3, "What is the Portal Life Cycle?"
- Section 9.4, "Creating a WebCenter Portal: Framework Application"
- Section 9.5, "Understanding the Features of a WebCenter Portal: Framework Application"
- Section 9.6, "Developing Your Portal's Look and Feel"
- Section 9.7, "Changing Default Portal Preferences"
- Section 9.8, "Using Iterative and Round-Trip Development Techniques"
- Section 9.9, "What is the Application Sources Folder?"
- Section 9.10, "Running and Testing a Portal"
- Section 9.11, "WebCenter Portals at Runtime"
- Section 9.12, "How are WebCenter Portal Files Organized?"
- Section 9.13, "Changing the Default Home Page and Login/Logout Target Pages"
- Section 9.14, "Basic Portal Development Tasks"

9.1 What is a Framework Application?

A WebCenter Portal: Framework application is a portal. From an end user perspective, a portal is a web site with **pages** that are organized by tabs or some other form(s) of **navigation**. Each page can contain a nesting of **sub-pages** that are related hierarchically. Any page can contain multiple portlets, task flows, or other elements, giving users access to different information and tools in a single place. An administrator can modify a portal at runtime by, for example, adding new pages or changing the look and feel. If authorized through **delegated administration**, individual users can modify their view of a portal as well.

Figure 9–1 shows an example portal. It includes navigation (tabs, bread crumbs, links), content (portlets, task flows, images), administration features, and so on.

Figure 9–1 Example WebCenter Portal: Framework Application



Portal end users may also see **personalized** views of a portal page. A personalized view is a view that has been modified with content that is based on information collected about that user. For example, a site that sells books might "remember" that you recently purchased several detective novels, and based on that information, recommend several other similar kinds of detective novels when you next log in to that site. See Chapter 67, "Personalizing WebCenter Portal Applications."

A portal presents diverse content and applications to users through a consistent, unified web-based interface. Portal administrators and users can customize portals, and content can be presented based on user preferences or personalization. Each portal is associated with a web application that contains all of the resources required to run portals on the web.

As you can see, some key elements of portals include pages and sub-pages, navigation, delegated administration and other security features, runtime customization, and personalization.

Portals also typically include features like:

- **Search** – Enterprise and web-based search facilities
- **Content Integration** – Creation, management, and delivery of content to the portal through integration with an enterprise content management system, like Oracle WebCenter Content.
- **Single Sign-On** – Allows users to log on once for all applications within the portal.

- **Look and Feel Management** – Allows developers to create custom page templates and users to apply different skins to customize their portal experience. The portal includes tools to ensure a consistent look and feel across the portal.

9.2 What is Oracle WebCenter Portal: Framework?

Oracle WebCenter Portal: Framework adds a number of portal-specific features to a Framework application, like page hierarchies and navigation models, as explained in this section.

- Section 9.2.1, "How Do I Add Oracle WebCenter Portal: Framework to a Project?"
- Section 9.2.2, "What Features Does the Oracle WebCenter Portal: Framework Provide?"

9.2.1 How Do I Add Oracle WebCenter Portal: Framework to a Project?

If you create a Framework application using the WebCenter Portal: Framework Application template, the framework is automatically included. No further steps are required. See also Section 5.2, "Creating a Framework Application."

Note: Migrating a non-portal Framework application to a Framework application is not supported.

9.2.2 What Features Does the Oracle WebCenter Portal: Framework Provide?

Oracle WebCenter Portal: Framework adds a number of portal-specific features to a Framework application. These features are included by default when you create a new application using the WebCenter Portal: Framework Application template. See also Section 5.1, "What Are Application Templates?."

The basic features provided by the framework include:

- **Page hierarchies** – Traditional ADF page security allows you to control page security on a page-by-page basis. However, portals typically include many pages – in some cases hundreds of pages – and it's not always practical to specify access to individual pages. The page hierarchy organizes pages into a tree structure, with a parent-child relationship between pages. This hierarchical structure allows the convenient propagation or inheritance of security settings from pages to sub pages. See Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."
- **Navigation models** – A navigation model provides the back-end data to the navigation user interface that is displayed in the portal. Navigation models are highly flexible, with a wide set of APIs that control navigation to pages, task flows, external sites, portlets, and specific content items. These APIs let you control all aspects of the way the navigation user interface is rendered. For example, you can add a tree navigation UI by including a simple EL expression in a portal page or page template. WebCenter also provides runtime tooling for configuring and modifying navigation models. See Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry."
- **Navigation Registries** – A navigation *registry* defines the set of elements that a user can add to a navigation at runtime using the Resource Manager. See also Chapter 17, "Adding and Using the Resource Manager" and Chapter 11, "Building a Navigation Model for Your Portal."

- **Delegated administration** – Delegated administration provides a mechanism for securing portal resources based on user roles. For example, you can allow users in one role (managers, for instance) to access all portal features, but deny certain features to users in another role (employees, for instance). Page hierarchies are the primary container for delegated administration. In both the design time and runtime environments, you apply delegated administration to a page hierarchy, and the specific security assignments are automatically propagated down through the hierarchy through pages and sub pages. See Chapter 69, "Securing Your WebCenter Portal: Framework Application."
- **Customization** – Users can customize their own portal pages by, for example, adding or removing portlets, changing skins, or rearranging the portal layout. See Section 17, "Adding and Using the Resource Manager."
- **Page templates** – Page templates allow you to define entire page layouts and apply them to pages to create a consistent layout across the portal. When you create pages using a template, they all inherit the defined layout. For more information, see Chapter 12, "Designing the Look and Feel of Your Portal." See also "Using Page Templates" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.
- **Catalogs and the Catalog Registry** – Catalogs are arbitrary collections of task flows, portlets, content items, and other elements that can be added to an application at runtime. A catalog *registry* defines the set of items that are available for inclusion in your resource catalog. See Chapter 16, "Creating and Managing Resource Catalogs."
- **Skins** – Skins define the colors, fonts, images, and some dimensional details like heights and widths, of your application components to present a consistent look and feel across the portal. See Chapter 15, "Creating and Managing Skins."
- **Portal Preferences** – You can change default preferences for a portal, including default navigation model, page template, and other components. See Section 9.7, "Changing Default Portal Preferences."

JDeveloper includes tooling for portal development. This tooling includes editors for creating and modifying page hierarchies, navigation models, catalogs and the catalog registry, page templates, page styles, skins, and delegated administration. In addition, runtime tools are provided to allow end users and administrators to perform some of these functions. See also Section 9.11.2, "Runtime WebCenter Portal Administration Console."

9.3 What is the Portal Life Cycle?

The portal life cycle describes the process of creating a portal from development through deployment to a production server. Many actors participate in the life cycle including software developers, content modelers, content contributors, IT administrators, portal site administrators. For detailed information on managing all the stages of the portal life cycle, see Chapter 10, "Understanding the WebCenter Portal Life Cycle."

9.4 Creating a WebCenter Portal: Framework Application

Use the WebCenter Portal: Framework Application template to create a new framework application in JDeveloper. Application templates are a standard JDeveloper feature, and the portal template simply configures the application with the required portal components and features.

Tip: The Framework application template is basically identical to the Framework application template in earlier versions of WebCenter, but with certain portal-specific features added in. In addition to these portal features, framework applications include all of the features you'll find in a traditional Framework application, like task flows, data controls, security features, ADF Faces layout components, and so on.

The basic steps for creating a framework application are described in Section 5, "Creating Framework and Portlet Producer Applications."

9.5 Understanding the Features of a WebCenter Portal: Framework Application

This section discusses some of the main portal features you'll need to know about.

- Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy"
- Section 9.5.2, "Securing Your Portal Pages"
- Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry"
- Section 9.5.4, "Understanding Resource Catalogs and the Catalog Registry"
- Section 9.5.5, "Other Design Time Folders"
- Section 9.5.6, "Tips for More Advanced Development Use Cases"

9.5.1 Understanding Pages, Page Templates, and the Portal Page Hierarchy

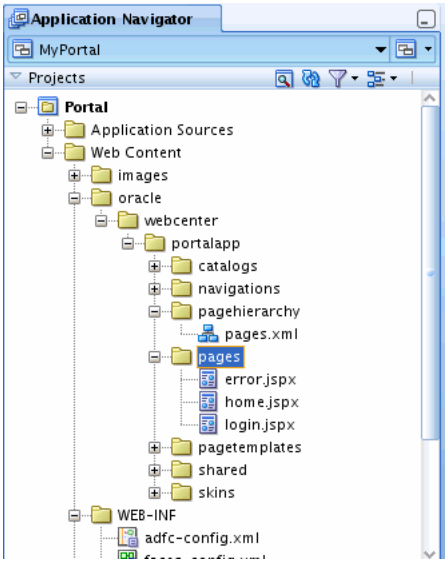
This section introduces three related pieces of the Oracle WebCenter Portal: Framework: pages, page templates, and page hierarchies.

A portal consists of one or more *pages*, and pages play a crucial role in a portal's structure and organization. Generally, a page is a container for one or more entities like task flows, portlets, and content. Pages also typically include a navigation interface, like a navigation tree, tabs, or bread crumbs. In a Framework application, you can find a set of default pages in the Portal project under the folder `oracle/webcenter/portalapp/pages`, as shown in Figure 9–2. The placement of these pages is for organizational purposes, and new pages do not necessarily need to be created under the pages folder.

Note: For important information on the organization of files in Framework applications, see Section 9.12, "How are WebCenter Portal Files Organized?."

Note: For pages to be used in the page hierarchy, they must be located under `oracle/webcenter/portalapp`.

Figure 9–2 Contents of the pages Folder

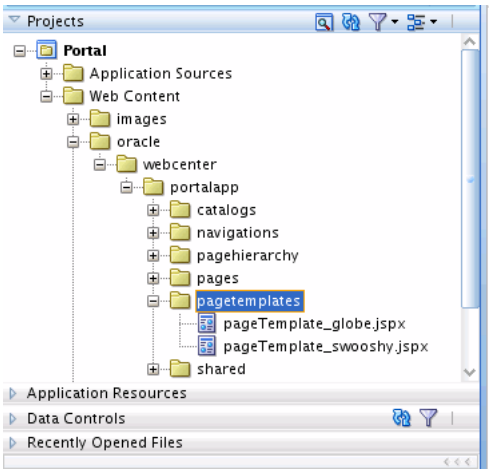


A *page template* lets you specify view elements that you wish to be common to all of your pages. A page template is a JSPX file that includes ADF layout components and other elements. Typically, page templates define a page layout, with headers, footers, and content areas. In addition, the page template usually specifies the positioning and style of the navigation UI for your pages. For example, if you want all of your pages to include the same set of tabs, or if you want each page to have a tree navigation panel and bread crumbs, you can add these components to the template.

> **Tip:** Page templates are referenced by the pages that use them. If you change the underlying page template, all the pages that reference the template automatically inherit the change.

You can find the page template folder in `oracle/webcenter/portalapp/pagetemplates`, as shown in Figure 9–3. Notice that the folder contains two default page templates that are included out of the box.

Figure 9–3 Contents of the pagetemplates Folder

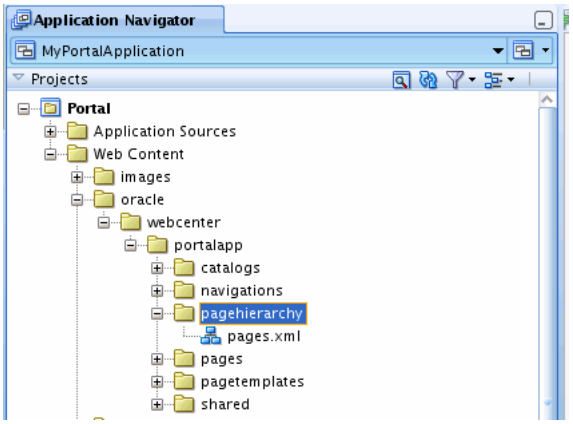


A *page hierarchy* is a logical structure that arranges pages through a set of parent-child relationships, where any page can have one or more sub pages. This hierarchical

model not only helps define the overall structure of the portal, but also allows child pages to inherit the security policies specified by their parent. The ability for pages to propagate security policies down the hierarchy is particularly convenient for very large portals with dozens or even hundreds of pages: you don't have to set security for every page. Note that when you create a new page, its security policy is automatically set based on where it resides in the page hierarchy.

You can find the page hierarchy files in the `oracle/webcenter/portalapp/pagehierarchy` folder, as shown in Figure 9–4.

Figure 9–4 Contents of the pagehierarchy Folder



> **Note:** The page hierarchy files are XML files that contain the metadata that define the page hierarchy structure. It is not recommended that you attempt to edit these files directly; rather, use the Page Hierarchy editor to create and work with page hierarchies, as explained below.

A basic page hierarchy model is included with applications created from the WebCenter Portal: Framework Application template. You can modify the page hierarchy with the Page Hierarchy editor in JDeveloper. To access this editor:

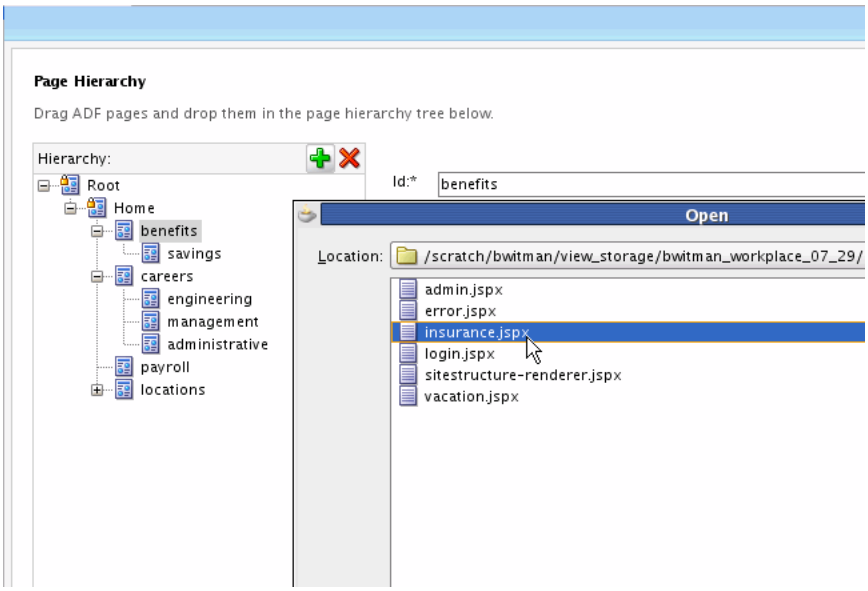
1. Open the portal project in the Application Navigator, as shown in Figure 9–4.
2. Right-click the `pagehierarchy` folder.
3. Select **Edit Page Hierarchy** from the menu.

> **Tip:** You can also bring up the Page Hierarchy editor by right-clicking a portal page file (JSPX file) and selecting **Edit Page Hierarchy** from the menu or by opening a page hierarchy file directly.

After you have created pages for your portal, use the editor to build up the site structure. You can drag and drop pages (one or several at a time) from anywhere in the `oracle/webcenter/portalapp` folder in the Application Navigator directly into the Page Hierarchy editor and position them in the hierarchy as you wish.

Alternatively, you can use the **Add** button located above the hierarchy pane. Clicking this button pops up a dialog that allows you to choose a page from within your project. This page is added as a child of the page that is currently selected in the editor, as shown in Figure 9–5.

Figure 9–5 Adding a Page to the Page Hierarchy



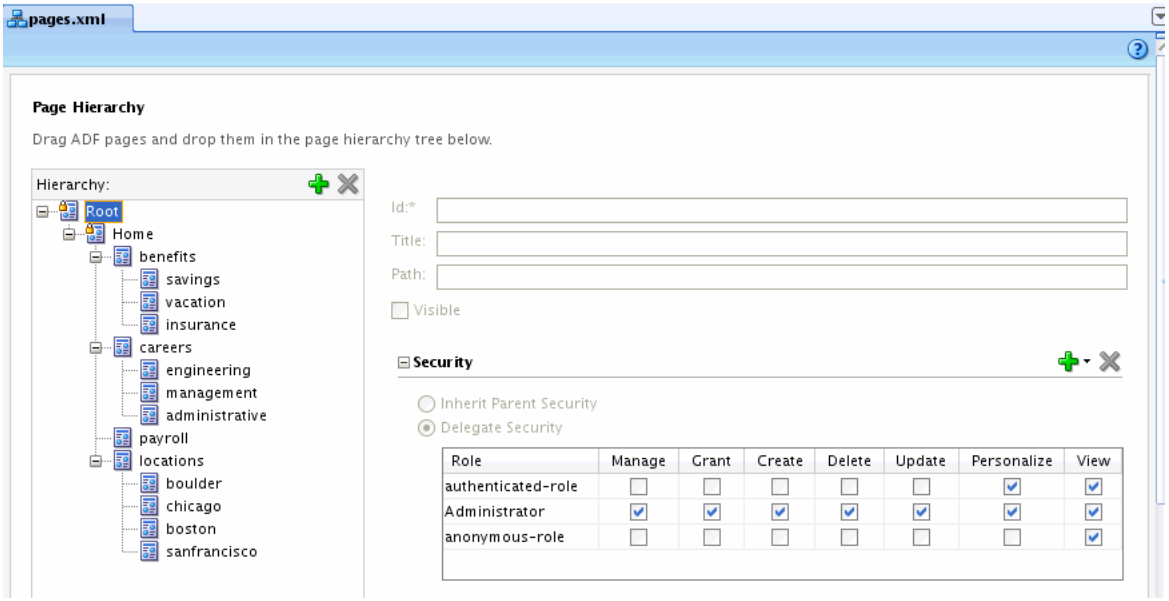
In the editor, you can rearrange page nodes within the tree display simply by dragging and dropping them. For large hierarchies, you may also find it useful to use the Change Parent dialog by right-clicking a page in the hierarchy and choosing **Change Parent Page**. This dialog allows you to choose a new parent for the selected page from a separate window, instead of having to scroll through a hierarchy that may be too large to view all at once.

> **Tip:** Right-click a node in the Page Hierarchy editor and select **Go to Page** to open the associated JSPX (portal page) file.

> **Note:** For each new level you create in the page hierarchy, a `pagehierarchy/pages/*Pages.xml` file is created. These files contain a reference to the pages in that level. For more information, see Section 69.6.2, "Building a Page Hierarchy."

Figure 9–6 shows the structure of a simple human resources site that includes top level pages Benefits, Careers, Payroll, and Location. The Benefits, Careers, and Location pages each have multiple child pages.

Figure 9–6 Page Hierarchy Editor



Notice how this hierarchy is realized when you run the portal, shown in Figure 9–7. In this example, the pages use the default page template, which includes a navigation bar. The bar is rendered as tab-like links across the page. Because this navigation UI was added to the page template, it will appear consistently on all pages that use the template. In addition to the default navigation component, you could add a navigation tree, bread crumbs, or other styles of navigation UI to the page template. In many cases, the template will include two or more navigation components.

For more information on page templates, see Section 12, "Designing the Look and Feel of Your Portal." For more information on the default navigation model and on creating navigation models and user interfaces, see Section 11, "Building a Navigation Model for Your Portal."

Figure 9–7 Portal Navigation Bar



For more information on page hierarchies, see Section 69.6.2, "Building a Page Hierarchy."

9.5.2 Securing Your Portal Pages

Note: See also Section 5.6, "How Is a Framework application Security Configured by Default?."

At design time, you can set up security for your portal's pages in the Page Hierarchy editor. As mentioned previously, the page hierarchy defines the structural relationships between pages (parent-child) and it allows for child pages to inherit the security policies of their parents.

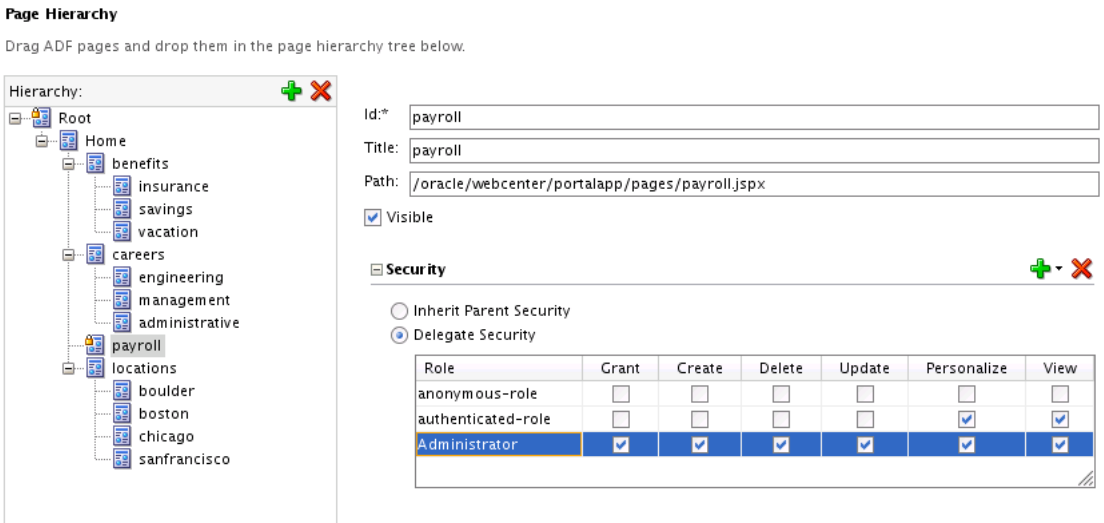
> **Note:** You can secure pages either through the ADF page security model or through the hierarchical WebCenter Portal page security model. When a page is added to the page hierarchy, it implies that it's secured through the hierarchical security model. See also Section 69.6, "Using the Page Hierarchy Security Editor."

As shown in Figure 9–8, the Page Hierarchy editor includes a Security section that lets you specify role-based security policies for pages. The security interface lets you decide whether you want the policies to be inherited or delegated. If you specify inheritance, then the security policies for that page will be inherited from its parent page. If you delegate security, then that policy will override the parent's security and the policy will be propagated to all child pages that are inheriting security.

> **Tip:** A small padlock icon appears next to pages for which the inherited security settings are overwritten. In other words, you'll see this icon to indicate pages that have delegated security set on them.

For example, in Figure 9–8, the Payroll page is configured so that Administrators have full access to the page, while authenticated users can only view and personalize the page. Anonymous users cannot access the page at all. Later, in Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry," we'll discuss how the security policy set for a page also affects the appearance of the navigation UI. In this example, because the Payroll page is only available to Administrators and authenticated users, the navigation UI will only show the Payroll page link to authenticated users and Administrators. For all other users, the Payroll link will not show up at all.

Figure 9–8 Security Setting for the Payroll Page



9.5.3 Understanding the Navigation Model and the Navigation Registry

It's important to distinguish the navigation model from the page hierarchy. First, the page hierarchy specifies a parent-child relationship between pages. As discussed previously, pages can have one or more child pages. Secondly, the page hierarchy allows for security policies to be inherited from parent pages to child pages, or sub pages, down through the hierarchy. Note that page hierarchy only specifies a relationship between pages; other resources, like task flows, portlets, and external links cannot be included in a page hierarchy.

The page hierarchy provides a security inheritance model that allows administrators to easily control security on portals that have a very large number of pages. For example, if your portal has 100 pages, you can specify security policies on the root of the page hierarchy, and all 100 pages will inherit those policies. Then, if you wish to modify the settings on any page below the root, you can do that, and those settings will propagate to any of that page's sub-pages. You can also override the inherited settings and apply security settings to individual pages anywhere in the hierarchy.

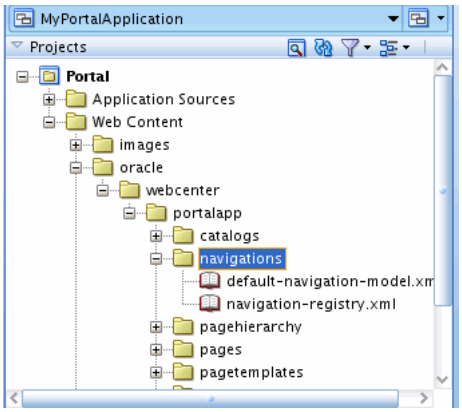
The navigation model, by contrast, defines a navigational structure, keeps track of navigational data, and provides that data to the view layer of the application. While a page hierarchy only consists of pages, a navigation model can consist of a variety of resources, like task flows, pages, external links, and others.

It's important to note that the navigation model is aware of the security policies that have been applied to the elements (pages, links, task flows, and so on) that the navigation model controls. If the authenticated user is not authorized to see a particular page, the navigation model, by default, hides any navigational links to that page. For any resources that do not have security defined in the portal, like external links, the developer has to control visibility manually. One way to do this is to add an EL expression to the page or page template to control visibility (show/hide) of the resource. For more information, see Chapter 14, "Visualizing Your Portal Navigation."

For example, if your portal pages include a tree navigation UI, that UI asks the navigation model questions like: what pages or other elements can the user navigate to and what is the current navigational state of the portal. The first question informs which links the UI can display. The second informs the UI how to display those links. If the navigation view is a tree, the model tells the UI, for example, which page is currently selected. This allows the tree UI to display the node representing that page correctly, as an open folder for example, or as a selected page.

In JDeveloper, use the Navigation Editor to modify the navigation model. To open the editor, double-click the file `default-navigation-model.xml` in the `navigations` folder, shown in Figure 9–9.

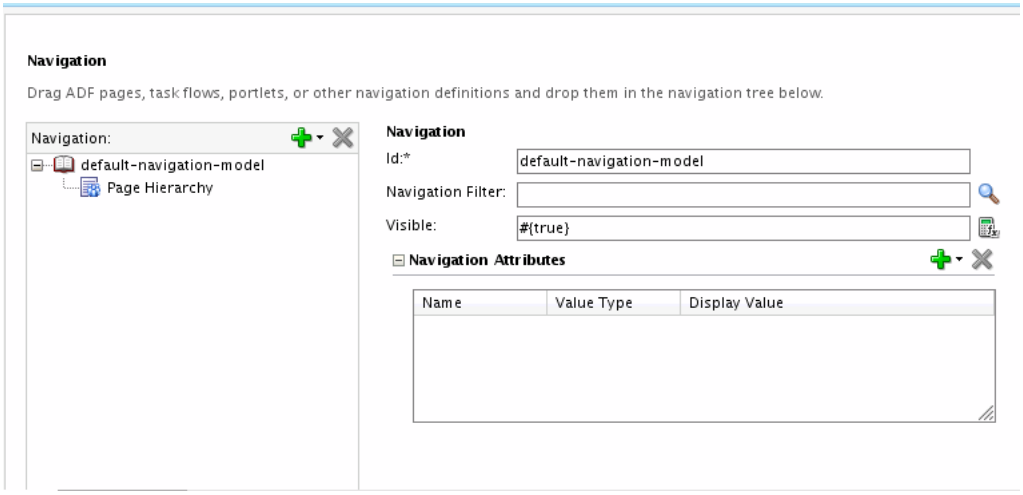
Figure 9–9 Contents of the navigations Folder



The `navigation-registry.xml` file is used by the Resource Manager when a user creates or edits a navigation model. The registry defines the superset of all items that are available to be included in a model that you create or edit. You can create multiple navigation models for an application, but an application can only have one navigation registry file. For more information on the navigation model, the Navigation Editor, and the registry, see Chapter 11, "Building a Navigation Model for Your Portal."

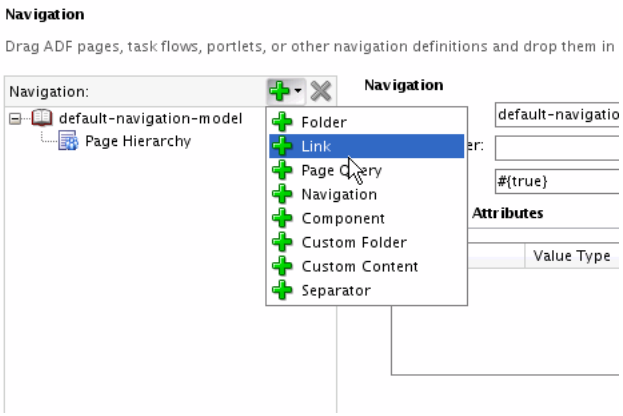
The default navigation model, shown in Figure 9–10, includes one node called Page Hierarchy. By default, this node points to the default page hierarchy file, `pages.xml`, in the pagehierarchy folder. This node tells the navigation model to refer to the page hierarchy's structure and communicate that structure to the navigation UI so that it can be rendered. As seen previously in Figure 9–7, the default UI displays the page hierarchy as tab-like links. The information provided by the model allows each link to display its sub-page links in a drop-down menu whenever the mouse pointer hovers over it.

Figure 9–10 The Default Navigation Model



The navigation model can include several kinds of navigation elements. As we've seen, the model can include both page hierarchies and single pages. But you can also create navigation to other elements, like links to external web pages or specific content.

Figure 9–11 Adding a Link to the Navigation Model



For more information, see Chapter 11, "Building a Navigation Model for Your Portal."

9.5.4 Understanding Resource Catalogs and the Catalog Registry

A catalog specifies a collection of an otherwise unrelated group of elements, like layout components, task flows, portlets, documents, and others, that an authorized user can add to a portal at runtime. Oracle WebCenter Portal's Composer uses catalogs at runtime to determine which elements an authorized user can add to a portal page.

The Framework application template includes a default resource catalog (`default-catalog.xml`) and a catalog registry file (`catalog-registry.xml`). These files are located in `oracle/webcenter/portalapp/catalogs`.

When a user edits pages at runtime using Composer, the default catalog file specifies all of the items that are available to be added to a page. The catalog registry file is used by the Resource Manager when a user creates or edits a resource catalog. The registry defines the superset of all items that are available to be included in a catalog that you create or edit.

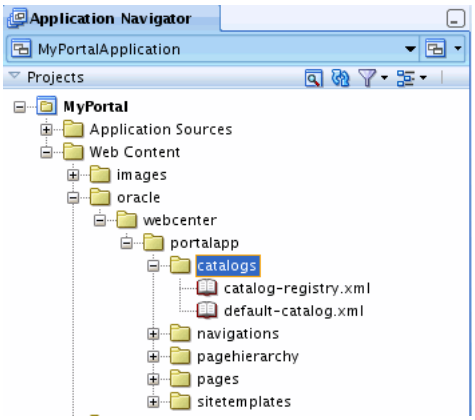
You can create multiple catalogs for an application, but an application can only have one catalog registry file. For more information, see Chapter 16, "Creating and Managing Resource Catalogs."

At development time, you can edit existing catalogs using the Resource Catalog editor. To open the editor, open the catalog file (for example, `default-catalog.xml`) in the `catalogs` folder in Application Navigator, as shown in Figure 9–12.

To create a new catalog, select **New** from the File menu. In the New Gallery dialog, select **Portal** under the Web Tier node. Then, select **Application Resource Catalog** and click **OK**. Use the Create Application Resource Catalog dialog to create the new catalog.

> **Tip:** You can also edit navigation registries using the Resource Catalog editor.

Figure 9–12 Contents of the catalogs Folder



The Resource Catalog editor is shown in Figure 9–13.

Figure 9–13 Resource Catalog Editor in JDeveloper

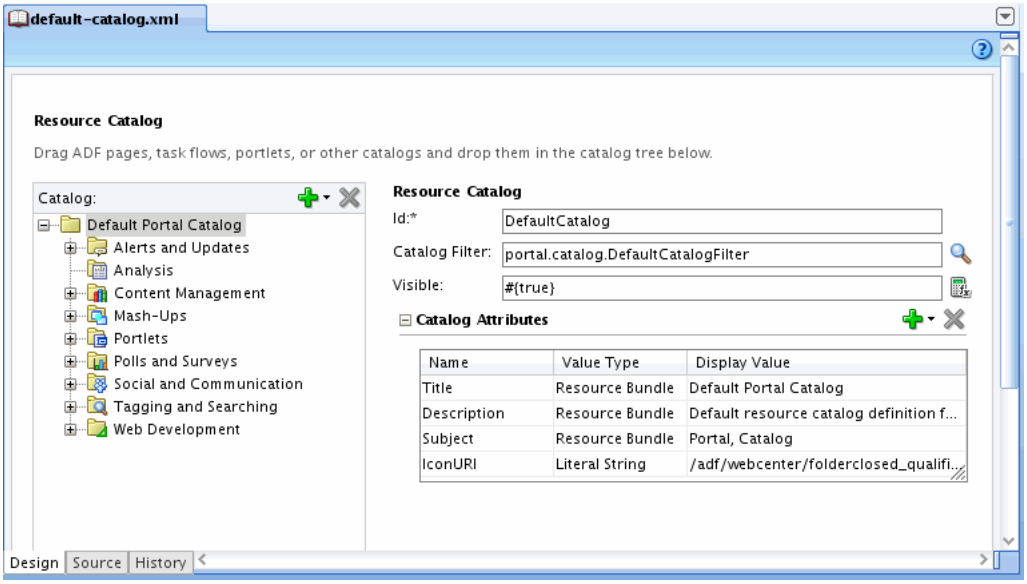
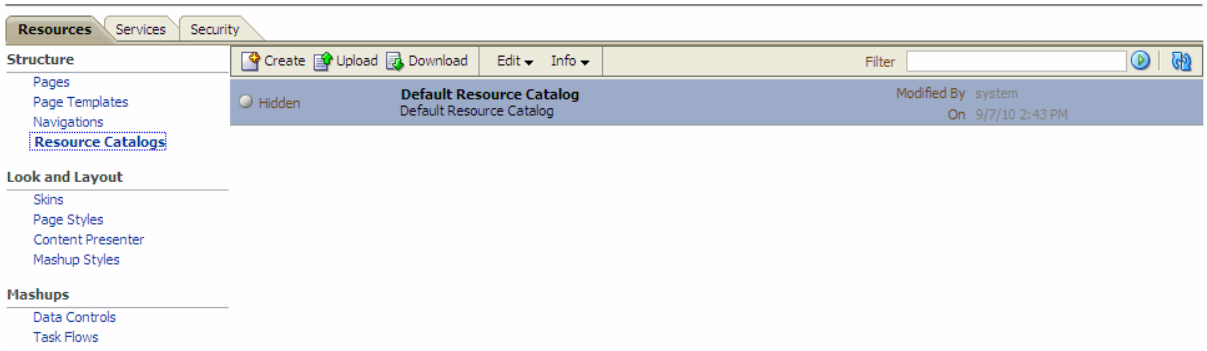


Figure 9–14 shows the resource catalog editor in the runtime administration tool.

Figure 9–14 Resource Catalogs Editor at Runtime



For example you can create separate catalogs for each of several departments.

Tip: Individual catalog entries can be shown or hidden based on user roles.

9.5.5 Other Design Time Folders

Other folders you'll see in the Application Navigator at design time are:

- **images** – Contains images that are referenced by the page template.
- **WEB-INF** – Contains configuration files used by the web application. These files are generally typical files found in all ADF-based web applications.
- **Page Flows** – Another standard ADF folder that contains the `adfc-config.xml` config file.

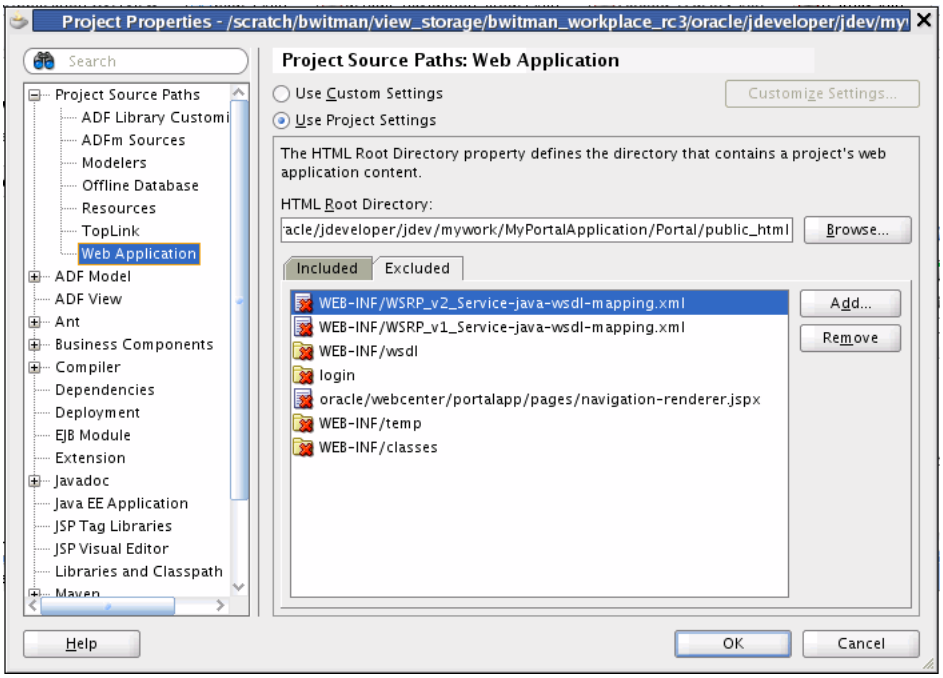
9.5.6 Tips for More Advanced Development Use Cases

By default, a few files are excluded from view in the Application Navigator. These files are excluded because it is unlikely you'll ever need to edit them. However, because there are some use cases where you might want to edit them, you need to be able to add them back to the JDeveloper UI.

To locate the excluded files:

1. Right-click the project folder and select **Project Properties** from the menu.
2. In the Project Properties dialog, select **Web Application** under the Project Source Paths node.
3. Select the Excluded tab, as shown in Figure 9–15.

Figure 9–15 The Exclude Tab



4. Select an element from the excluded list and click **Remove** to remove it from the excluded list. As a result, the element is automatically included in the appropriate folder in the Application Navigator.

As an example, the `navigation-renderer.jspx` file is used to render resources in the context of a page template. If you wanted to change the way an external URL is displayed within a page, you would first remove the `navigation-renderer.jspx` file from the excluded list in order for it to appear under the `pages` folder in the Application Navigator. Then, you could open the page and edit it as desired.

9.6 Developing Your Portal's Look and Feel

You can design the look and feel by adding certain presentation elements, such as banners, navigations, and footers, around the content area. Once you settle on a look and feel that is suitable for your portal structure, you can save these settings as a page template, which can be used to create a portal. In addition, you may want use the same settings across pages in your portal. With the use of page templates and page styles, you can achieve some consistency across pages in your portal. See Chapter 12, "Designing the Look and Feel of Your Portal."

You can also develop custom skins for your portal. A skin is a style sheet based on the CSS 3.0 syntax specified in one place for an entire application. Instead of providing a style sheet for each component in your application or inserting a style sheet on each page, you can create one skin for the entire application. Every component automatically uses the styles as described by the skin. When you use a skin, you do not have to make design-time changes to portal pages to change their appearance. See Chapter 15, "Creating and Managing Skins."

Although the basic look and feel design can be created in JDeveloper at design time, the Resource Manager enables administrators and users with the appropriate privileges to continue developing the portal's look and feel after the application has been deployed. For example, the Resource Manager lets you add and remove pages, add navigation user interfaces, and change the page templates, skins, and page styles. For more information about these and other Resource Manager features, see Chapter 17, "Adding and Using the Resource Manager."

9.7 Changing Default Portal Preferences

A Framework application includes a set of preferences that specify certain default portal components. This section describes these preferences and explains how to change their default values.

9.7.1 What Are the Default Portal Preferences?

Table 9–1 lists the portal preferences and their default settings.

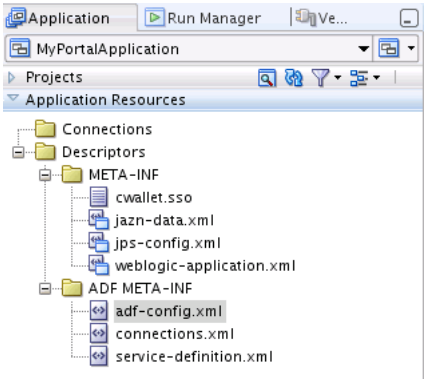
Table 9–1 Default Preferences

Preference	Default Setting
Navigation Model	`/oracle/webcenter/portalapp/navigations/default-navigation-model.xml`
Resource Catalog	`/oracle/webcenter/portalapp/catalogs/default-catalog.xml`
Page Template	`/oracle/webcenter/portalapp/pagetemplates/pageTemplate_globe.jspx`
Navigation Renderer	`/oracle/webcenter/portalapp/pages/navigation-renderer.jspx`
Skin	`portal`

9.7.2 How to Change the Default Preferences at Design Time

To change the default preferences at design time, directly edit the `adf-config.xml` file. To locate this file in JDeveloper, open the **Application Resources** part of the Application Navigator. Then, open the **Descriptors** folder and the **ADF META-INF** folder, as shown in Figure 9–16.

Figure 9–16 Location of the adf-config.xml File in JDeveloper



For example, to change the default navigation model file, edit the value of this preference:

```
<portal:preference id="oracle.webcenter.portalapp.navigation.model"
   desc="Default Navigation Model"
   value="/oracle/webcenter/portalapp/navigations/default-navigation-model.xml"
   resourceType="navigation" display="true"
/>
```

Preferences can also be changed at runtime under the **Configurations** tab of the Administration page. From this page, you can configure the default page template, skin, resource catalog, and navigation component in a runtime application. For details, see "Configuring Application Defaults" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

9.8 Using Iterative and Round-Trip Development Techniques

Iterative development is a productivity feature that helps speed up the development process. Iterative development lets you make and save changes to your Framework application in JDeveloper while the app is running on the Integrated WebLogic Server and immediately see the effect of those changes simply by refreshing the current page in your browser. The iterative development feature is enabled by default in a framework application. For more information, see Section 1.5.3, "Enabling Iterative Development."

Round-trip development refers to features and techniques that allow you to retrieve resources from a deployed, runtime portal back to JDeveloper for maintenance or enhancement. After modifying a resource in JDeveloper, you can use the Resource Manager to upload the resource back to the deployed portal. WebCenter Portal's round-trip development features provide a simple, convenient way to modify portal resources without redeploying the entire application.

For more information on round-trip development, see Section 17.3, "Enabling Runtime Administration of Your Portal" and Section 17.4, "Enabling Round-Trip Development of Resources."

9.9 What is the Application Sources Folder?

The Application Sources folder appears as one of two top-level portal project folders generated by JDeveloper when you create a portal. In addition to any Java classes you write for your application, Application Sources includes:

- **oracle.webcenter.portalapp** – Includes XML definition files for pages, page templates, page catalogs, navigations. A page definition file specifies ADF bindings, page parameters, and permission settings. Page definition files specify various mappings and bindings that are used by pages and page templates. For example, the page definition file specifies the page's parent and child pages, if any. The file also specifies security policy information, like the operations that are permitted on the page. Default page definition files are created automatically when you add a portal page to a page hierarchy.
- **portal** – Includes resource bundles, source code, and other Java artifacts.
- **META-INF** – Includes files for specifying data bindings and page template metadata.

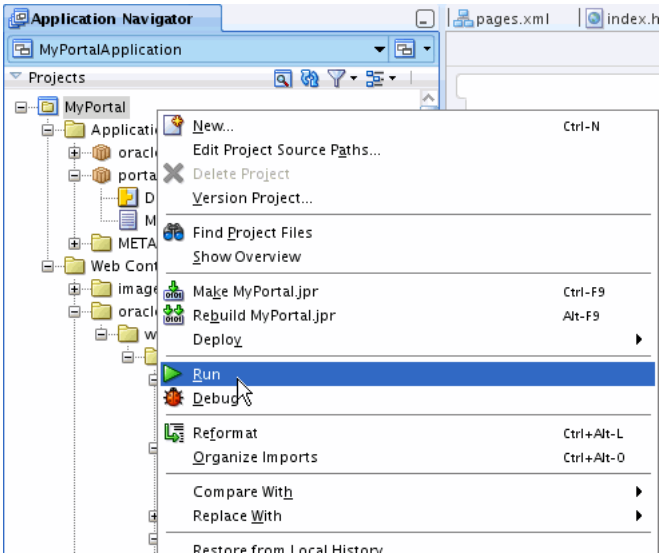
9.10 Running and Testing a Portal

JDeveloper provides several ways to run an application:

- Right-click the portal project name in the Application Manager and select **Run**, as shown in Figure 9–17. If the server is not running, this action starts the server and (re)deploys the application.

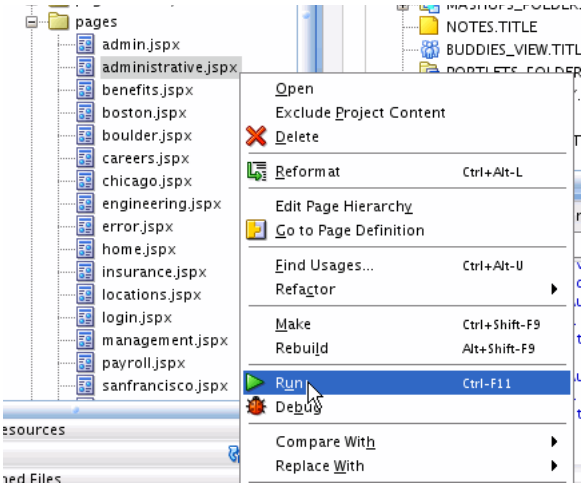
  Tip: You can change the default portal home page. For details, see Section 9.13, "Changing the Default Home Page and Login/Logout Target Pages."

Figure 9–17 Running the Portal



- Right-click a page in the pages folder in the Application Manager and select **Run**, as shown in Figure 9–17. If the server is not running, this action starts the server, (re)deploys the application, and displays the selected page.

Figure 9–18 Running a Page



- You can also run a portal by selecting **Run** <*portal name*> from the Run menu.

9.11 WebCenter Portals at Runtime

Oracle WebCenter Portal: Framework provides a number of interesting runtime features.

- Section 9.11.1, "Preserving Runtime Customizations on the Integrated WebLogic Server"
- Section 9.11.2, "Runtime WebCenter Portal Administration Console"
- Section 9.11.3, "How Security Settings Affect the Runtime Portal?"
- Section 9.11.4, "Runtime Editing of Portal Resources"

9.11.1 Preserving Runtime Customizations on the Integrated WebLogic Server

If you are using the Integrated WebLogic Server in a development environment (running the portal through JDeveloper) any changes you make to the portal at runtime (using the Resource Manager) are discarded upon redeployment by default. For example, if you use the Resource Manager to make changes like adding entitlements to a page, changing the layout, or modifying the navigation model, these changes will not be preserved the next time you redeploy the application. For more information on the Resource Manager, see Chapter 17, "Adding and Using the Resource Manager."

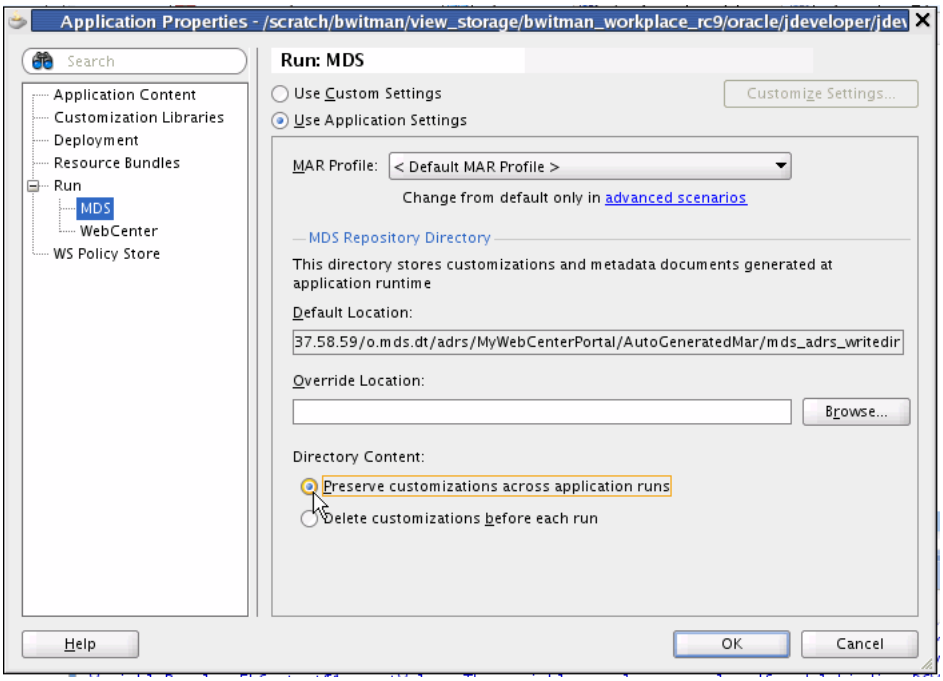
Note: The information in this section only applies to a portal that is running with the Integrated WebLogic Server in a development environment. When you deploy the portal to a production environment, runtime changes are never discarded.

Note: Customizations are not preserved by default as a convenience for developers working in the JDeveloper environment. When you modify a file at runtime, a new version of the file is written to the MDS write directory. This new version then takes precedence over the version in JDeveloper. At that point, if you change a setting in JDeveloper and click Refresh in your browser, the change will not show up. Therefore, while you're working in JDeveloper, it's much more convenient and natural not to preserve runtime customizations.

It is possible to change the default behavior, and preserve runtime customizations between runs. For example, you may wish to do this to enable certain testing scenarios. To allow customizations to be preserved between application deployments (runs):

1. Select **Application Properties** from the Application Menu.
2. In the Application Properties dialog, select **MDS** under the Run node.
3. Under Directory Content, select **Preserve customizations across application runs** to disable the default, which is to discard customizations before each run. See Figure 9–19.

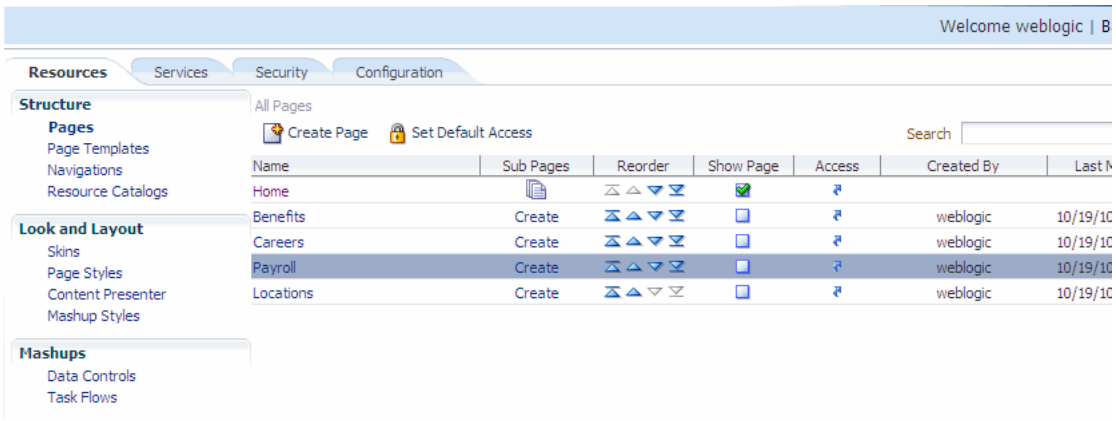
Figure 9–19 Selecting Preserve Customizations Across Application Runs



9.11.2 Runtime WebCenter Portal Administration Console

The WebCenter Portal Administration Console lets you work with resources, services, security, and portal configurations. The WebCenter Portal Administration Console is located at this URL: `http://<server>:<port>/<context_root>/admin`, and is shown in Figure 9–20.

Figure 9–20 The WebCenter Portal Administration Console



Tip: You can change the default URL for the WebCenter Portal Administration Console by editing the `<url-pattern>` attribute of the `<servlet-mapping>` element in the `web.xml` file. The default `<url-pattern>` is `admin`, but you can change it to something else if you want.

For more information on the Resource tab (the Resource Manager), see Section 9.11.4, "Runtime Editing of Portal Resources" and Chapter 17, "Adding and Using the Resource Manager." Security topics are discussed in Chapter 69, "Securing Your WebCenter Portal: Framework Application."

9.11.3 How Security Settings Affect the Runtime Portal?

Role based security policies control which pages, resources, and navigational elements visitors can see and manipulate (create, delete, update, and so on). The design time (JDeveloper) page editor lets you set these policies on pages or hierarchies of pages, as discussed in Section 9.5.2, "Securing Your Portal Pages." Oracle WebCenter Portal: Framework respects these security polices in the following ways:

- Pages (and certain other resources, like task flows) can only be viewed by users who are authorized to see them. The same principle holds for operations users can perform on pages, like Grant, Create, Delete, Update, and Personalize.
- Navigation to a resource (like a page or task flow) is hidden if that page is unavailable to the authenticated user. For example, if a user is not authorized to visit the **Payroll** page, the link to that page will not show up in any of the navigational user interfaces.
- Resources that are available within a Resource Catalog are adjusted based on security policies. If a user is not authorized to access a particular resource, such as a task flow, that resource will not appear in the Resource Catalog.

9.11.4 Runtime Editing of Portal Resources

The WebCenter Portal Administration Console includes a Resources tab that lets you work with several portal-specific features at runtime:

- Pages
- Page templates
- Navigation models

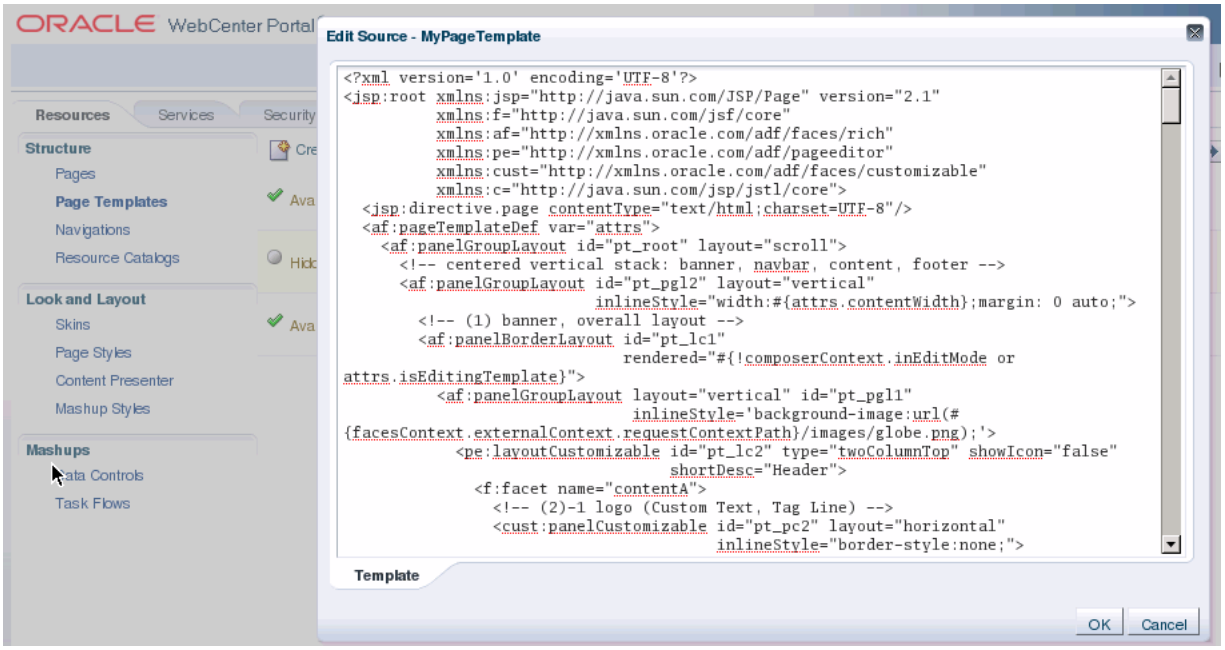
- Resource Catalogs
- Skins
- Page styles
- Content Presenter display templates
- Mashup styles
- Data controls
- Task flows

Using the Resource Manager, portal users can also download resources, or an entire application, from the runtime environment, edit them in JDeveloper, and then upload them back into the deployed application. For more information on downloading and uploading resources, see Section 17.3, "Enabling Runtime Administration of Your Portal" and Section 17.4, "Enabling Round-Trip Development of Resources."

The Edit Source feature in the Resource Manager lets you edit the source code of resources in the runtime application. For example, if you upload a portal resource (like a page template), you can then edit it directly in the Resource Manager. Just select the resource, and choose Edit Source from the Edit menu. A source editing window appears, as shown in Figure 9–21.

Note: Some resources, like the `default-navigation-model`, are not editable. If you want to edit these resources, make a copy first and edit the copy.

Figure 9–21 Editing a Page Template Source File at Runtime



9.12 How are WebCenter Portal Files Organized?

When you run through the Framework application wizard, a large number of project artifacts are configured and installed in the project directory. JDeveloper presents a

streamlined view of your project. You can also view your portal project directly on your filesystem.

- Section 9.12.1, "Understanding the Organization of a WebCenter Portal: Framework Application"
- Section 9.12.2, "Viewing Your Portal Project on the Filesystem"
- Section 9.12.3, "Viewing Your Portal Project in JDeveloper"

9.12.1 Understanding the Organization of a WebCenter Portal: Framework Application

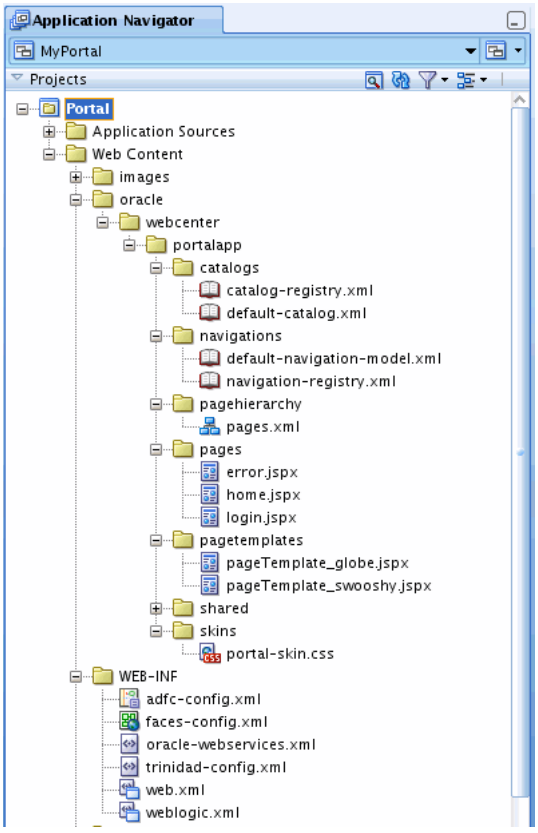
This section explains how a Framework application is organized, and then why it is organized the way it is.

9.12.1.1 How is a WebCenter Portal: Framework Application Organized?

When you create a new Framework application in JDeveloper, a large number of files are automatically placed in the project. You can see these files organized in the Application Navigator view in JDeveloper.

Looking at the Application Navigator, the first thing you will notice is that a framework application consists of one project, which is called (by default) **Portal**, as shown in Figure 9–22. You can, of course, change this name when you create the application or anytime afterwards. Most of the files in a project are placed in the `<application_root>/<project_root>/public_html/oracle/webcenter/portalapp` directory.

Figure 9–22 WebCenter Portal: Framework Application in the Application Navigator



9.12.1.2 Why Is a WebCenter Portal: Framework Application Organized the Way It Is?

It is important to understand the following distinctions between files that are stored in your application:

> **Note:** It is important to understand these distinctions because some files, like XML files, you might not expect to find under `public_html`. (Because in a standard web application, most files under this directory are directly URL accessible from a browser once the application is deployed.) Developers must keep in mind that portal files require this particular structure; you must not attempt to create portal artifacts in other locations.

- Files located in the `<application_root>/<project_root>/public_html/oracle/webcenter/portalapp` directory:
  - are deployed to the Metadata Services (MDS) repository.
  - can be registered as portal resources and managed at runtime with the Resource Manager.
  - are generally secured using WebCenter Permissions (if they are portal resources).
- Files located elsewhere under `<application_root>/<project_root>/public_html`:
  - are deployed to the application WAR file.
  - cannot be registered as portal resources and therefore cannot be managed with the Resource Manager.
  - are secured using the native permission class of the artifact.

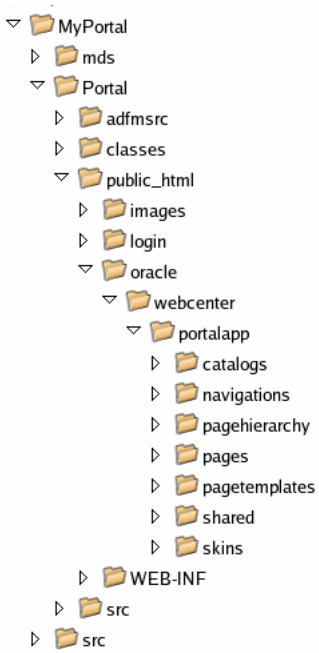
It is important to understand that a Framework application is structured this way specifically so that you can make informed decisions when you create your own pages. In some cases, specifically with JSPX pages, you might not want to create a page under `oracle/webcenter/portalapp`. It depends on the intended use of the page in your application.

9.12.2 Viewing Your Portal Project on the Filesystem

The simplest way to locate your project files on the file system is to select a file or folder in the Application Navigator and then select **Copy Path** from the Edit menu. This function places the path to the file or folder on the clipboard, which you can then paste into a command shell or file browser.

Figure 9–23 shows the filesystem organization of a sample Framework application. Many of the project's files are organized under the `public_html` folder.

Figure 9–23 Sample WebCenter Portal: Framework Application on the Filesystem



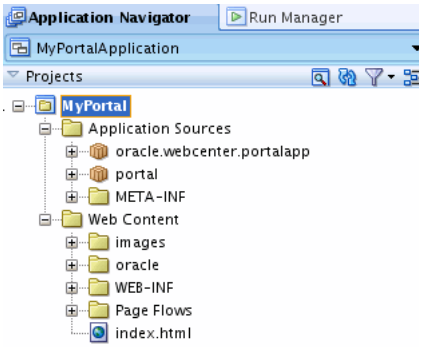
As you will see, JDeveloper presents a somewhat different, more streamlined view of your project, as discussed in the next section, Section 9.12.3, "Viewing Your Portal Project in JDeveloper."

9.12.3 Viewing Your Portal Project in JDeveloper

The intent of the project view in JDeveloper is to present the project files that a developer is likely to work with. These files include pages, page hierarchies, navigation models, page templates, catalogs, XML configuration files, Java source files, images, and so on.

In JDeveloper, portal projects are organized into two folders: Application Sources and Web Content, as shown in Figure 9–24.

Figure 9–24 Organization of a Portal Project in JDeveloper

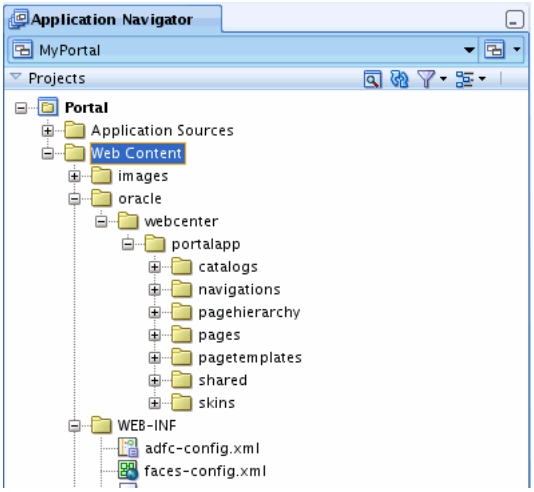


The **Application Sources** folder is primarily a repository for source code and page definition files. See also Section 9.9, "What is the Application Sources Folder?."

The **Web Content** folder contains all of the files that make up your web project, like pages, page hierarchies, navigation models, and so on. These are the files that you will actively create and modify as you develop your Framework application.

Figure 9–25 shows the basic structure of the Web Content folder. Note that many of the files you will create and modify are located in the `oracle/webcenter/portalapp` sub folder. This folder's contents – catalogs, navigations, page hierarchy, pages, skins, and pagetemplates – make up the basic components of a portal. For detailed information on these portal components, see Section 9.12.1, "Understanding the Organization of a WebCenter Portal: Framework Application."

Figure 9–25 The Web Content Folder



Tip: If you prefer to view the local file system hierarchy in the Application Navigator, click the **Navigator Display Options** icon in the Projects panel and select **Group by Directory**.

The **PortalWebAssets** project includes static application resources like HTML and image files. By separating the static resources into a separate project, it is possible to deploy those resources to a dedicated server. For more information, see Section 5.4, "Understanding the PortalWebAssets Project.".

9.13 Changing the Default Home Page and Login/Logout Target Pages

The file `oracle/portalapp/pages/home.jspx` is the default home page for a new Framework application. This section explains how to change this default home page to another page or any navigable resource like a URL, portlet, or task flow.

- Section 9.13.1, "Understanding How the Home Page is Specified"
- Section 9.13.2, "How to Change the Default Home Page"
- Section 9.13.3, "How is the Default Index Page Specified?"
- Section 9.13.4, "Specifying the Target Page After a Login or Logout"

9.13.1 Understanding How the Home Page is Specified

The default Framework application includes an `index.html` file, which is located in the portal project directory. This index file contains the following redirect statement:

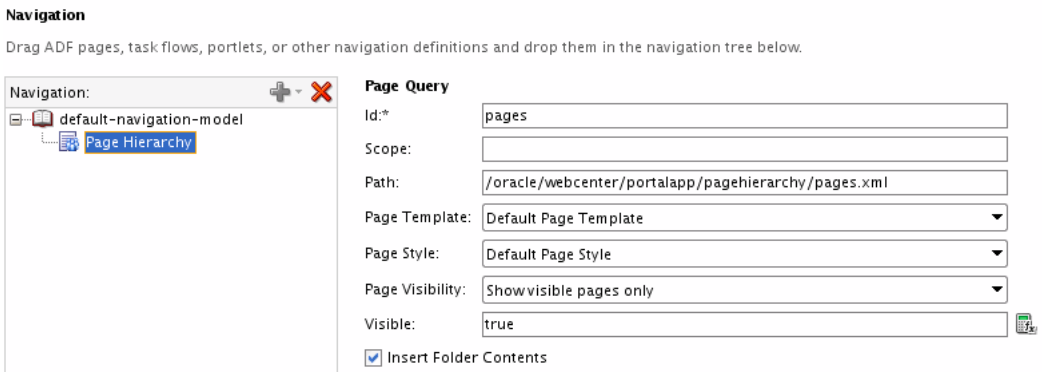
```
<meta http-equiv="refresh" content="0;url=./faces/pages_home" />
```

Note: The redirect statement in `index.html` refers to the portal navigation model, not to a page as you might expect. Pointing the redirect to the navigation model allows the `currentSelection` attribute to be set properly, which allows the current selection to be highlighted in the user interface.

The URL element `pages_home` refers to the "home" page specified in the page hierarchy. (The prefix "`pages_`" is added to the folder name simply to ensure a unique ID.) As Figure 9–26 shows, the **Insert Folder Contents** checkbox is selected in the Navigation dialog. The **Insert Folder Contents** option replaces the specified folder ("`home`" in this case) with the *contents* of the folder. Therefore, in this example, the portal looks for a folder called "`home`" in the page hierarchy. Note too that Page Hierarchy is specified as an element in the navigation model.

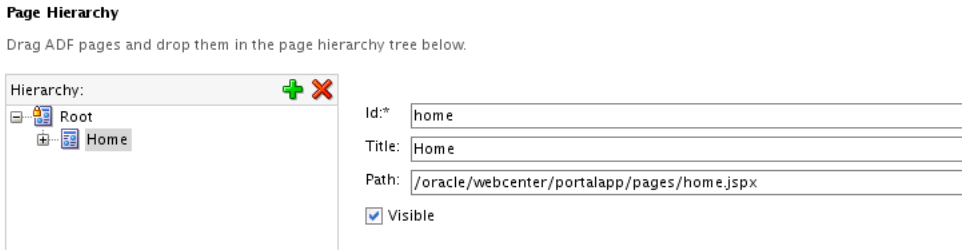
To help you understand how the default portal home page is specified, note that in the navigation model a default node called `pages` is created by default. This node's navigation path points to the page hierarchy file, `pages.xml`, as shown in Figure 9–26.

Figure 9–26 The Default Navigation Model



Upon navigating to `pages.xml`, the node called `home` is located, and that node's path is `/oracle/webcenter/portalapp/pages/home.jspx`, as shown in Figure 9–27.

Figure 9–27 The Default Page Hierarchy



9.13.2 How to Change the Default Home Page

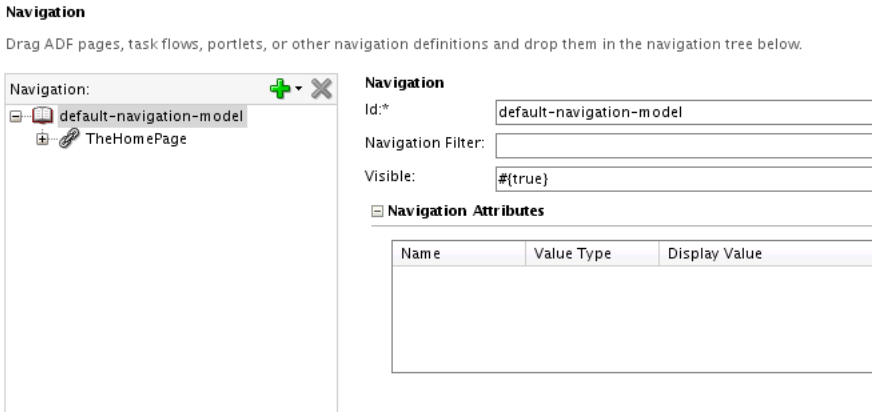
To change the default home page, simply change the redirect statement in `index.html` to the pretty URL of another resource that is referenced in the navigation

model. For example, you might create a new home page and redirect to it – a common use case. Simply specify the new page to be your portal's home page by changing the redirect in `index.html` to the new page's pretty URL. (You can drag the new page in the navigation model to create a page link element.) For example, if the link ID in the navigation model is `TheHomePage`:

```
<meta http-equiv="refresh" content="0;url=./faces/TheHomePage" />
```

where `./faces/TheHomePage` is the pretty URL that points to the link element defined in the navigation model.

Figure 9–28 Navigation Model with New Home Page Element

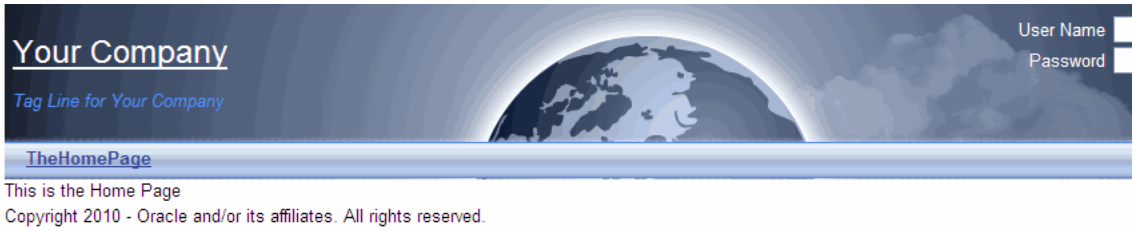


Using this example, if access the portal in a browser using this URL:

```
http://myserver:myport/MyPortalApp-Portal-context-root
```

the portal will render a home page that contains the content from the new home page, as shown in Figure 9–29.

Figure 9–29 The Oracle Website Displays as the Content for the Portal Home Page



9.13.3 How is the Default Index Page Specified?

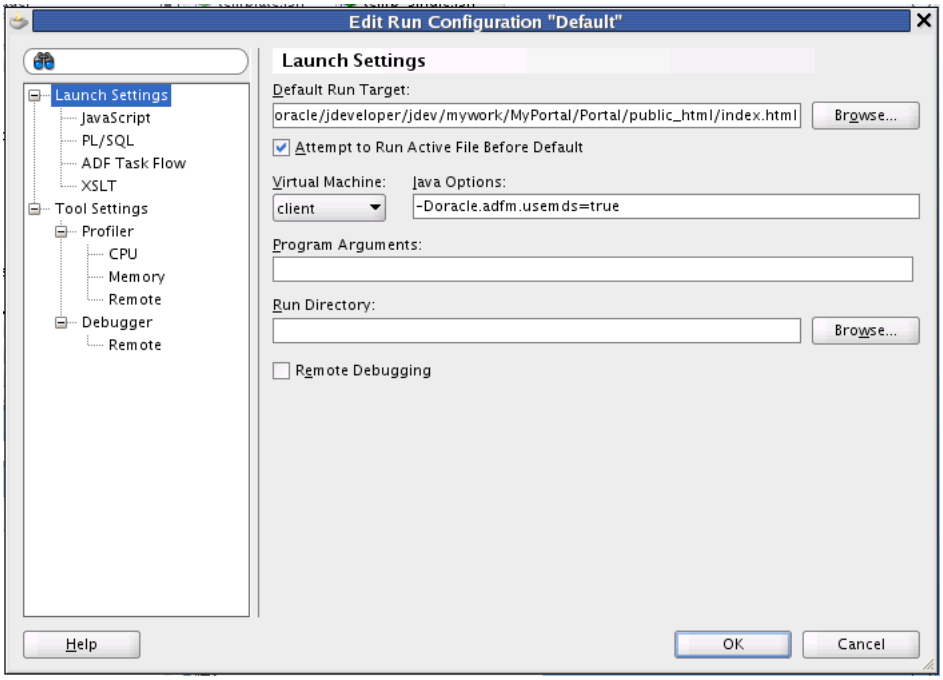
The default index page for the Integrated Weblogic Server instance is `index.html`. A new portal project includes this file, by default, in the `Web Content` folder.

To change the default index file, `index.html` to another file, you have two choices. One way to change this default index file is to edit the `<welcome-file-list>` element in the `web.xml` file:

```
<welcome-file-list>
    <welcome-file>/index.html</welcome-file>
</welcome-file-list>
```

Another way to change the default index page is to use the Edit Run Configuration "Default" dialog box. To access this dialog, open the project properties dialog, then, select the **Run/Debug/Profile** option. Click **Edit** to bring up the Edit Run Configuration "Default" dialog where you can change the default index page, as shown in Figure 9–30.

Figure 9–30 Edit Run Configuration "Default" Dialog



9.13.4 Specifying the Target Page After a Login or Logout

To specify the target page after a login or logout, edit the `<navigation-rule>` element in the `WebContent/WEB-INF/faces-config.xml` configuration file in your portal project, as shown in Example 9–1. You can change the target pages to any other navigation resource's pretty URL or Faces page.

Example 9–1 Specifying the Target Page for Login and Logout

```
<navigation-rule>
    <from-view-id>*</from-view-id>
    <navigation-case>
      <from-outcome>login_success</from-outcome>
      <to-view-id>/pages_home</to-view-id>
      <redirect/>
    </navigation-case>
    <navigation-case>
      <from-outcome>logout_success</from-outcome>
      <to-view-id>/pages_home</to-view-id>
      <redirect/>
    </navigation-case>
</navigation-rule>
```

For example, to redirect the user to a post-logout page follow these steps:

1. Create the post-logout page and add content to it. For example:

   `/oracle/webcenter/portalapp/pages/postlogout.jspx`

2. Modify the navigation rule in `faces-config.xml`. For example:

```
<navigation-case>
  <from-outcome>logout_success</from-outcome>
  <to-view-id>/oracle/webcenter/portalapp/pages/postlogout.jspx</to-view-id>
  <redirect/>
</navigation-case>
```

After a successful logout, the user is redirected to the specified page.

9.14 Basic Portal Development Tasks

This section lists some of the basic tasks involved with developing Framework applications.

- Set up an integrated team development environment with source control, a common database, and common content repository.
- Create a Framework application using the WebCenter Portal: Framework Application template.
- Design the overall structure of your portal, sketching out the top level pages and sub pages that will comprise your portal.
- Consider security for your portal. Decide which pages users will be allowed to visit. Create appropriate roles to accommodate these decisions.
- Begin thinking about your portal's overall look and feel, and begin working on a page template.
- Consider which WebCenter Portal services you want to include in your portal. For example, do you want to add a wiki or a blog or an activity stream? See Section 2.4, "Introducing Oracle WebCenter Portal: Services."
- Create pages based on your overall portal structure. The pages can be blank at this point, but creating them now allows you to begin assembling the page hierarchy.
- Create a page hierarchy using the Page Hierarchy editor.
- Create a navigation model. Think about the kinds of resources you want users to be able to navigate to. Examples include pages, external URLs, task flows, and content folders.
- Add navigation UI to the page template. Oracle WebCenter Portal: Framework provides several options for navigation UI. The most commonly used option is to use EL directly in the page template. WebCenter also provides navigation task flows and a Java API for navigation.
- Begin applying security policies to the page hierarchy. Start by applying policies to the root node of the hierarchy. Those policies will be inherited by all other pages in the hierarchy. Then, fine-tune the security settings on individual pages or on sub-branches of the overall hierarchy.
- Work on the pages themselves, adding and configuring portlets, task flows, content, and other features.

10

Understanding the WebCenter Portal Life Cycle

This chapter discusses tasks, tools, and techniques for managing a WebCenter Portal: Framework application throughout its life cycle.

- Section 10.1, "What is the WebCenter Portal Life Cycle?"
- Section 10.2, "What Are the Major Life Cycle Tasks?"
- Section 10.3, "Who Participates in the Portal Life Cycle?"
- Section 10.4, "Understanding the Build and Test Environments"
- Section 10.5, "Understanding the Staging and Production Environments"
- Section 10.6, "Tools for Managing the Life Cycle"
- Section 10.7, "Configuring a Nightly Build Script"
- Section 10.8, "Setting Up a Staging or Production Environment for the First Time"
- Section 10.9, "Moving a WebCenter Portal: Framework Application to an Existing Environment"
- Section 10.10, "Deploying to Managed Servers"
- Section 10.11, "Deploying and Configuring the Application on Targeted Servers"
- Section 10.12, "Building and Deploying Personalization for WebCenter Portal Files"
- Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production"
- Section 10.14, "Propagating Content From Oracle WebCenter Content"
- Section 10.15, "Managing Security Through the Life Cycle"
- Section 10.16, "Migrating Portlet Preferences"
- Section 10.17, "Rolling Back Production Site Changes"

10.1 What is the WebCenter Portal Life Cycle?

The portal life cycle refers to the path a portal takes from development through production. The phases of the life cycle typically include development, testing, staging, and production. Each phase requires certain tasks to be performed. Some tasks are performed only once, like setting up a content repository. Others are performed more frequently, like nightly builds. The phases of the portal life cycle are described in Table 10–1.

Table 10–1 Portal Life Cycle Phases

Life Cycle Phase	Primary Actors/Roles	Description
Development	■ Developers ■ Content Modelers ■ Content Contributors	The development portal is primarily source control and file-based. Developers work locally in JDeveloper and deploy to the Integrated WebLogic server. The development portal typically employs test data and content. Some of the features that are developed in this phase of the life cycle include: ■ portlets ■ task flows ■ shared libraries ■ skins ■ navigation models ■ page templates ■ display templates ■ content models ■ data transfer and interportlet communication ■ security The code from the development environment is built (usually nightly) and deployed to a clean, independent, targeted environment. WebCenter provides a build script that can be adapted for this purpose. See Section 10.7, "Configuring a Nightly Build Script."

Table 10–1 (Cont.) Portal Life Cycle Phases

Life Cycle Phase	Primary Actors/Roles	Description
Testing	■ Developers ■ QA Engineers ■ IT Administrators	The development portal is built (usually nightly) and deployed to an independent testing environment. The test environment typically includes a Metadata Service (MDS) and policy store that are database-based, and a dedicated Oracle WebCenter Content instance. The testing environment may contain test data and test content that will not become part of the production portal. Portlet producers may be shared between the test and development environments. However, if the usage load is high, Oracle recommends that separate instances be created. See also Chapter 62, "Testing and Deploying Your Portlets."
Staging	■ Site Managers ■ IT Administrators ■ Content Contributors	The staging environment provides a stable environment where final configuration and testing takes place before the portal is moved to production. Content contributors add content and refine the portal structure. Typically, the staging environment includes a dedicated Oracle WebCenter Content server, as well as dedicated portlet producer server(s). The staging server is often maintained as a mirror of the production site.
Production	■ Site Managers ■ IT Administrators ■ Content Contributors	A production portal is live and available to end users. A portal in production can be modified with tools like the Resource Manager and Oracle WebCenter Portal's Composer. For instance, an administrator might add additional portlets to a portal or reconfigure the contents of a portal. Individual users with proper authorization can also customize their view. See *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. WebCenter provides a propagation tool for migrating metadata to the production environment. See Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production." Content is provisioned using Oracle WebCenter Content replication tools. See Section 10.14, "Propagating Content From Oracle WebCenter Content." WebCenter provides WLST commands for importing and exporting portlet preferences. For more information, see Section 10.16, "Migrating Portlet Preferences."

10.2 What Are the Major Life Cycle Tasks?

Each phase of the life cycle requires actors (developers, administrators, content contributors, and others) to perform certain tasks. This provides an overview of the kinds of tasks that are performed during each phase of the portal life cycle.

- Section 10.2.1, "One-Time Setup Tasks"
- Section 10.2.2, "Development Environment Tasks"
- Section 10.2.3, "Nightly Build Environment Tasks"
- Section 10.2.4, "Testing Environment Tasks"
- Section 10.2.5, "Stage Environment Tasks"
- Section 10.2.6, "Production Environment Tasks"

10.2.1 One-Time Setup Tasks

You must perform certain preparatory steps to set up the development, build/test, stage, and production environments. Table 10–2 provides a general list of these preliminary setup tasks and the environments to which they apply. See also Section 10.7, "Configuring a Nightly Build Script" and Section 10.8, "Setting Up a Staging or Production Environment for the First Time."

Table 10–2 Typical One-Time Setup Tasks

Setup Task	Development	Build/Test	Stage	Production
Install JDeveloper	Yes	No	No	No
Install Oracle WebCenter Portal	Yes	Yes	Yes	Yes
Install Oracle WebLogic Server; create a domain and managed servers	No	Yes	Yes	Yes
Create required database schemas using RCU	No	Yes	Yes	Yes
Install and configure Oracle WebCenter Content	Yes	Yes	Yes	Yes
Install identity management components, such as Oracle Access Manager	No	Yes	Yes	Yes
Create the required Oracle Platform Security Services policies in the policy store	No	Yes	Yes	Yes
Create required user credentials in the credential store	No	Yes	Yes	Yes
Create connections to back end servers	Yes	Yes	Yes	Yes
Create build scripts	Yes	No	No	No
Create deploy and configure scripts	No	No	Yes	Yes
Integrate/configure Personalization for WebCenter Portal	Yes	Yes	Yes	Yes

* For more information on Personalization for WebCenter Portal, see "Managing Personalization for WebCenter Portal" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* and Chapter 67, "Personalizing WebCenter Portal Applications."

Another technique for setting up an environment for the first time is to clone an existing instance. Typically, this strategy is used to clone a production instance from a stage instance. For detailed information on this "test to production" technique, see "Moving from a Test to a Production Environment" in the *Oracle Application Server Administrator's Guide*.

10.2.2 Development Environment Tasks

In a development environment, each developer has a local JDeveloper instance that is connected to a source control system and a shared Oracle WebCenter Content repository. Developers store metadata and code in source control. Portlets are deployed independently to a producer server and consumed by the portal.

In a development environment, developers typically build and run the application locally using the Integrated WebLogic Server. Both MDS and the Policy Store are file-based in the local development environment.

For more information, see Chapter 1, "Developer's Quick Start Guide," Chapter 3, "Preparing Your Development Environment," and Chapter 4, "Working Productively in Teams."

10.2.3 Nightly Build Environment Tasks

On a daily basis, developers work in JDeveloper and run code changes locally on the Integrated WebLogic Server, and code changes are checked into a source control system.

If you wish to set up a build environment that produces a clean environment that is built each night, there are certain tasks that must be performed with each build, including:

- Creating schema (typically with Oracle Repository Creation Utility – RCU)
- Creating and setting up the domain
- Creating and configuring managed servers
- Deploying the application

WebCenter provides sample build scripts that you can modify and use for your environment. See Section 10.7, "Configuring a Nightly Build Script." Nightly builds are usually made available to people within the organization, like QA engineers, technical writers, managers, and others. See also Section 10.4, "Understanding the Build and Test Environments."

For information on integrating Personalization for WebCenter Portal into your development environment, see Section 67.2, "Integrating Personalization in Your Application."

10.2.4 Testing Environment Tasks

Usually, the testing environment is a mirror of the nightly build environment that is accessed by QA engineers. The tasks involved in providing a clean test environment are usually the same as creating a clean build environment. See also Section 10.4, "Understanding the Build and Test Environments."

10.2.5 Stage Environment Tasks

Runtime tools play a bigger role in the staging environment than in the development environment. Nonetheless, occasional updates from development will need to be deployed to the stage environment. To facilitate these more intermittent updates, WebCenter provides a set of deploy and configure scripts that you can copy and modify to suit your environment. The deploy and configure scripts isolate the information that is variable between environments, like the server names, ports, content management connections, and so on. For more information, see Section 10.5.1, "Provisioning the Staging Environment" and Section 10.11, "Deploying and Configuring the Application on Targeted Servers." See also Chapter 62, "Testing and Deploying Your Portlets."

For information on integrating and configuring WebCenter Personalization, see "Personalization for WebCenter Portal Prerequisites" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

10.2.6 Production Environment Tasks

When changes are tested and approved in the stage environment, they need to be pushed to the production environment. WebCenter provides a Propagation tool in the WebCenter Portal Administration Console that moves all portal metadata from the staging to the production server. For more information, see Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production."

WebCenter provides WLST commands for importing and exporting portlet preferences. For more information, see Section 10.16, "Migrating Portlet Preferences."

10.3 Who Participates in the Portal Life Cycle?

Many different people participate in the portal life cycle. In general, these people (the primary actors in Table 10–1) fall into one or more of these general roles:

- **Developer** – Uses JDeveloper and the build scripts described in Section 10.7, "Configuring a Nightly Build Script."
- **IT Administrator** – Uses the deploy and configuration scripts described in Section 10.11, "Deploying and Configuring the Application on Targeted Servers."
- **Site managers** – Have administrative privileges for the Framework application. Site managers use the WebCenter Portal Administration Console to modify the layout, content, and security settings of the portal. See also *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* and Section 10.15, "Managing Security Through the Life Cycle."
- **Content modelers** – Uses Oracle WebCenter Content's Site Studio designer to model content. Refer to Site Studio and Oracle WebCenter Content documentation for more information.
- **Content contributors** – Develops whatever content appears in or is available from the portal, including images, document files, video and audio content, and so on.

10.4 Understanding the Build and Test Environments

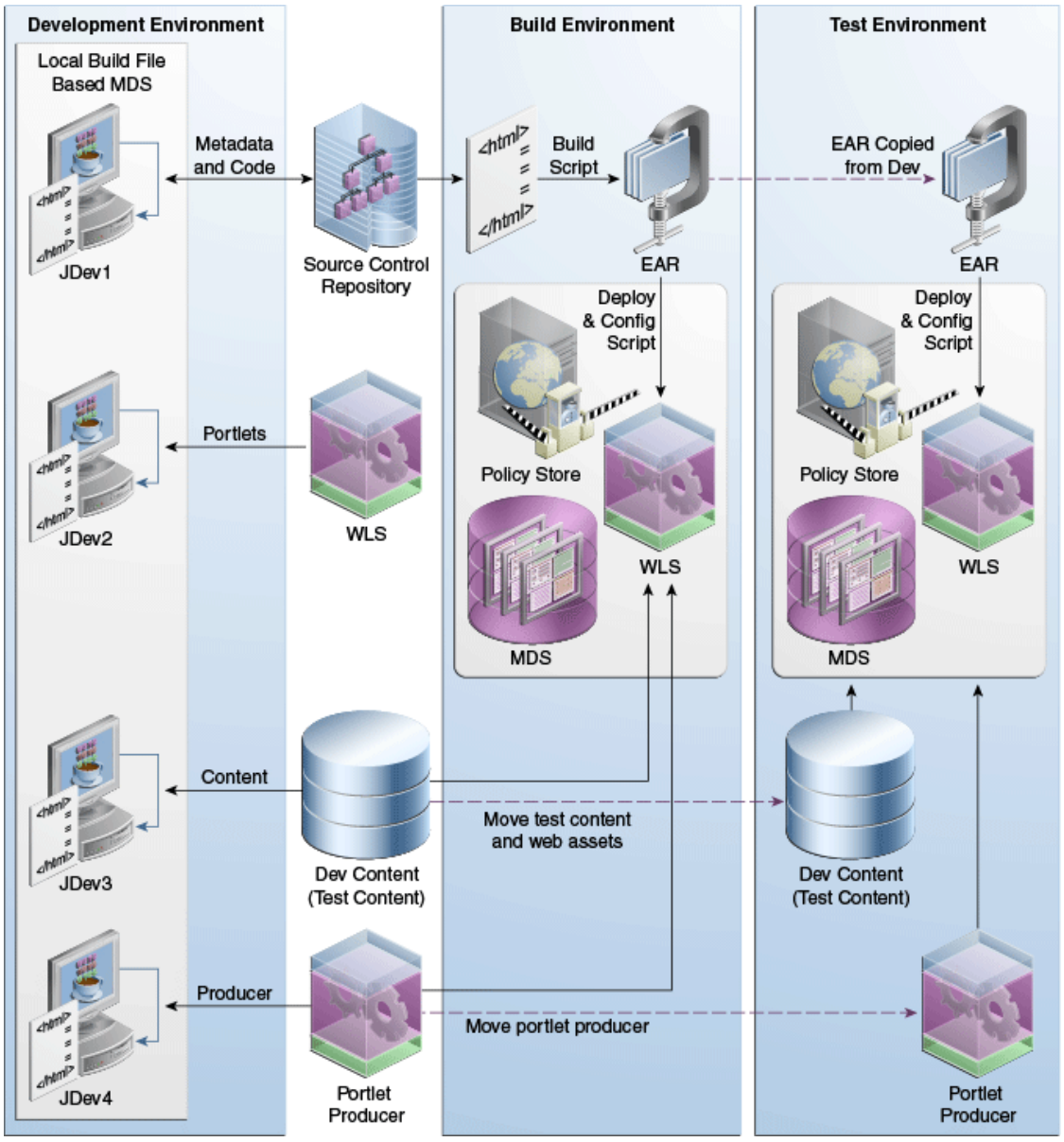
Although developers typically build and run locally, an automated nightly build environment is recommended. The build environment represents a clean deployment of the Framework application.

To set up the build environment, back end servers and connections must be installed and created once. For example, the build environment requires WebLogic Server instance, and typically includes access to an installed Oracle WebCenter Content instance. In addition, the build environment includes a database-based MDS and policy store.

Figure 10–1 illustrates the general flow from development to build to test environments.

Note: Figure 10–1 does not depict all possible portal features. For example, Personalization for WebCenter Portal is not depicted in the diagram. For more information, see Section 10.12, "Building and Deploying Personalization for WebCenter Portal Files."

Figure 10–1 Flow from Development to Build to Test Environments



A common practice is to use Ant-based build scripts to construct the EAR from source control and deploy it. The EAR includes metadata, code, and seeded policies. The build script also packages into the EAR metadata customizations made to portlets. See Section 10.7, "Configuring a Nightly Build Script." Some tasks performed by the build scripts include:

- creating schema (with RCU)
- setting up the managed server and domain
- building and deploying the EAR
- performing other setup tasks

The best practice is to use Oracle WebCenter Content archive tools to move Oracle WebCenter Content test content and web assets between the environments. See Section 10.14, "Propagating Content From Oracle WebCenter Content." For information on deploying portlets, see Chapter 62, "Testing and Deploying Your Portlets."

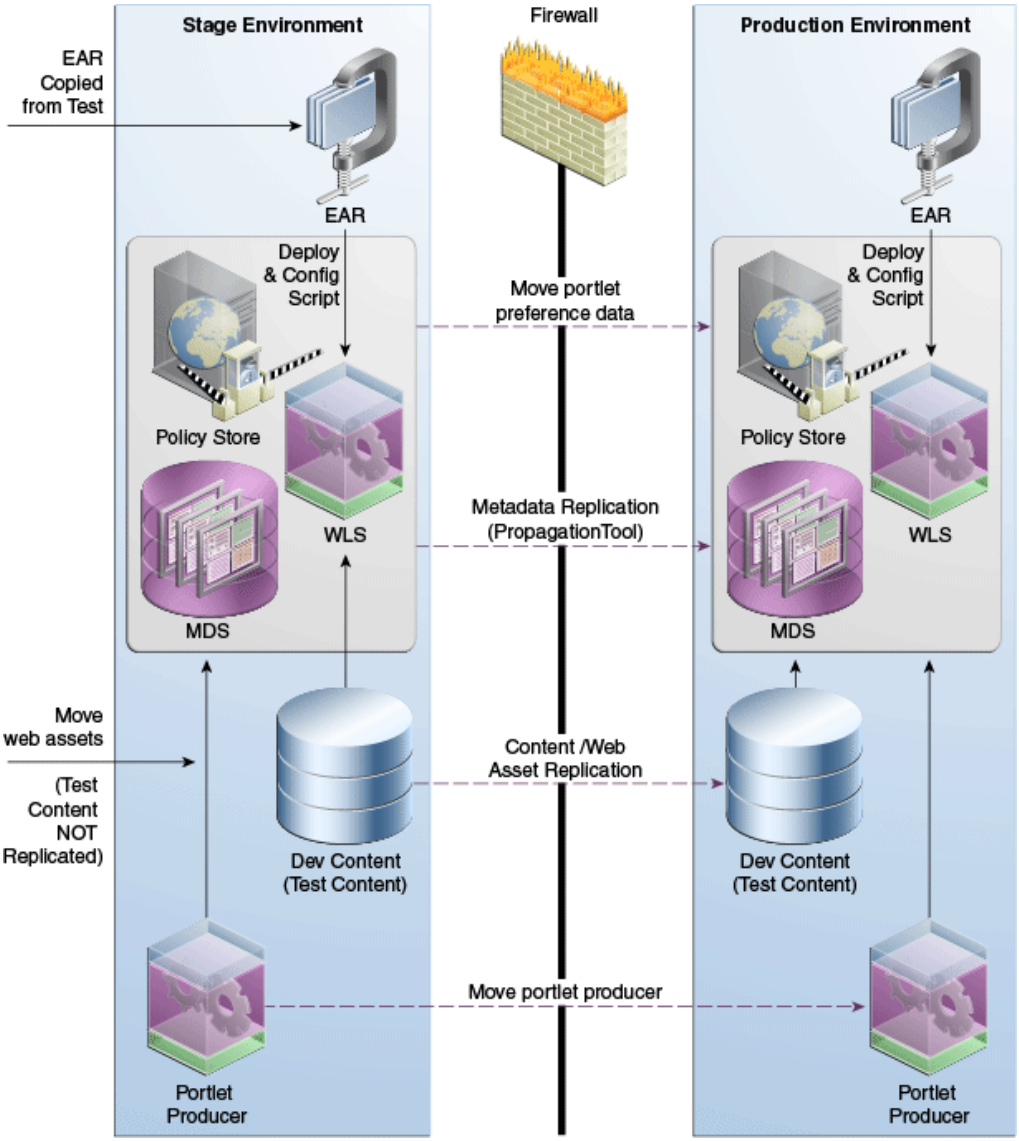
For information on integrating Personalization for WebCenter Portal into your development environment, see Section 67.2, "Integrating Personalization in Your Application."

10.5 Understanding the Staging and Production Environments

This section discusses the staging and production phases of the portal life cycle. Figure 10–2 illustrates the general flow from staging to production environments.

> **Note:** Figure 10–2 does not depict all possible portal features. For example, Personalization for WebCenter Portal is not depicted in the diagram. For more information, see Section 10.12, "Building and Deploying Personalization for WebCenter Portal Files."

Figure 10–2 Flow from Staging to Production Environments



10.5.1 Provisioning the Staging Environment

The staging environment provides a stable environment where final configuration and testing takes place before the portal is moved to production. Typically, the staging environment includes a dedicated Oracle WebCenter Content server, as well as dedicated portlet producer server(s). For a list of typical setup tasks, see Table 10–2.

If you are setting up the staging environment for the first time, see Section 10.8, "Setting Up a Staging or Production Environment for the First Time." For information on making incremental changes to the staging environment, see Section 10.11, "Deploying and Configuring the Application on Targeted Servers."

If you wish, you can move Oracle WebCenter Content content and web assets between the testing and staging environment using Oracle WebCenter Content archive tools. See Section 10.14, "Propagating Content From Oracle WebCenter Content." For information on migrating portlets, see Section 10.16, "Migrating Portlet Preferences."

For a complete list of requirements, dependencies, and options for WebCenter Personalization, see "Personalization for WebCenter Portal Prerequisites" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

10.5.2 Adding Content to the Staging Environment

Content developers can add content directly to the staging server using Oracle WebCenter Content content contribution tools and/or the Documents service. Content workflow features of Oracle WebCenter Content can be used to manage content approvals. WebCenter also provides runtime tools for creating and editing content. For example, you can:

- Edit existing Oracle WebCenter Content region data files
- Create new Oracle WebCenter Content region data files
- Edit existing HTML content
- Upload new images

In addition, the Documents service provides folder-based creation and editing of HTML content and other file types. See also Chapter 30, "Integrating the Documents Service."

10.5.3 Moving the Portal from Staging to Production

Once the staging environment is fully provisioned and tested, it can be moved to the production environment and made accessible to users. In a live production environment, you can make incremental updates to metadata, content, and web assets using automated scripts and/or replication techniques.

Typically, these updates are performed by a site manager rather than by individual content contributors. For more information, see Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production."

Any content model or region definition changes can be pushed to the production server by the site manager. Oracle WebCenter Content Replication can be used to replicate content between the stage and production server. See Section 10.14, "Propagating Content From Oracle WebCenter Content."

WebCenter provides WLST commands for importing and exporting portlet preferences. For more information, see Section 10.16, "Migrating Portlet Preferences."

10.6 Tools for Managing the Life Cycle

WebCenter Portal is built on top of JDeveloper and Oracle ADF, which provide several benefits in the life cycle management of your application:

- **Development framework:** JDeveloper and Oracle ADF provide the tools and framework you can use to build and update your application. Adding portlets, content, and customization capabilities to your Framework application is simply a matter of dragging and dropping the appropriate objects in either a source or WYSIWYG environment.
- **Iterative Development:** WebCenter provides an iterative development feature that greatly increases team productivity. See Section 1.5, "Preparing for Iterative Development."
- **Round-Trip Development:** Round-trip development refers to features and techniques that allow you to retrieve resources from a deployed, runtime portal back to JDeveloper for maintenance or enhancement. For more information on round-trip development, see Section 17.3, "Enabling Runtime Administration of Your Portal" and Section 17.4, "Enabling Round-Trip Development of Resources."
- **Enterprise deployment:** When you are ready to deploy your application to a production environment, you can use the deployment scripts provided by WebCenter. For more information, see Section 10.11, "Deploying and Configuring the Application on Targeted Servers" and Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production." For more information about deployment, see Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application."

  Note: WebCenter Portal: Services typically require some back end, such as Oracle WebCenter Portal's Discussions Server, to be available in the deployment environment.

- **Standards-based administration:** Browser-based tools enable administrators to deploy, configure, and manage Framework applications and WebCenter Portal: Services. In addition, tools built on industry standards-based JMX methods offer administrators granular control and monitoring mechanisms for health status, performance, and popularity. Tools for obtaining historical performance and status reporting over time (within a single Oracle Application Server context) are also provided. Framework application metrics are delivered using the familiar Application Server Control monitoring and management interface. For more information, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

10.7 Configuring a Nightly Build Script

When you create a Framework application, an Ant build script and properties file are created automatically for you in the main application directory on the filesystem. These files are:

- `build.xml`– Includes a standard set of targets for cleaning, compiling, building, and deploying the application. You can check this file into source control so that it is available to the developers.
- `build.properties` – Specifies the location of the JDeveloper workspace file (`.jws` file) for your application and the output folder for the application WAR file. Note that the `build.xml` file includes the `build.properties` file, which

enables `build.xml` to pick up these environment-specific variables. You can customize the properties file to match the setup and environment variables used in your build environment.

Before running the build script, be sure `JAVA_HOME` and `ANT_HOME` environment variables are set properly. For example:

```
JAVA_HOME = /path_to_your_java_installation/jdk
ANT_HOME = /path_to_your_ant_installation
```

To build and deploy your application, use the ant target `all`. For example:

```
ant all
```

If you are using Personalization for WebCenter Portal to provide personalization features for your portal, you must perform a manual update to the `build.xml` file. For details, see Section 10.12.1, "Building the WCPS MAR File."

10.8 Setting Up a Staging or Production Environment for the First Time

For detailed information on setting up and provisioning a staging or production environment for the first time, see "Moving Oracle WebCenter Portal to a New Production Environment," in the *Oracle Application Server Administrator's Guide*. See also Section 10.15, "Managing Security Through the Life Cycle" for information on moving security policies and credentials to an environment for the first time.

10.9 Moving a WebCenter Portal: Framework Application to an Existing Environment

In this scenario, you have a working production environment with Oracle WebCenter Portal installed and configured and you want to test changes in your applications or configuration before rolling those changes into the production environment. For example, you have modified existing security policies or configuration.

Oracle recommends that you use the deploy and configure scripts to update existing environment. Sample scripts are provided with WebCenter, and you are free to modify them to suit your environment. For detailed information, see Section 10.11, "Deploying and Configuring the Application on Targeted Servers" and Section 10.12, "Building and Deploying Personalization for WebCenter Portal Files."

WebCenter provides a Propagation tool that is built in to the WebCenter Portal Administration Console. The Propagation tool is for moving your application from the staging environment to the production environment. See Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production."

For information on moving Oracle WebCenter Content content, see Section 10.14, "Propagating Content From Oracle WebCenter Content."

For information on moving security policies and credentials, see Section 10.15, "Managing Security Through the Life Cycle."

For more information, see "Moving Oracle WebCenter Portal to an Existing Production Environment," in the *Oracle Application Server Administrator's Guide*.

10.10 Deploying to Managed Servers

Framework application developers working in JDeveloper typically deploy their application locally to the Integrated WebLogic Server. For other phases of the life cycle – testing, staging, and production – deployment to a suitable managed server is recommended. Or, for portlet development, you need to deploy to the Services Producer Managed Server. For more information on deploying to managed servers, see Section 70.3, "Deploying a Framework Application to a WebLogic Managed Server." See also "Templates for WebCenter Portal or Portlet Producer Applications" in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal* and "Creating and Provisioning a WebLogic Managed Server Instance" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

10.11 Deploying and Configuring the Application on Targeted Servers

During its life cycle, a typical portal is deployed to testing, staging, and production servers. This section describes a technique for deploying and configuring a Framework application on a targeted server.

- Section 10.11.1, "Introduction"
- Section 10.11.2, "Using the Deploy and Configure Script"

10.11.1 Introduction

WebCenter provides a configurable script that allows you to easily deploy and configure your application to these server instances. The build and deploy script takes a simple properties file parameter that specifies the server and connection information for the targeted server. You only need to create this properties file one time for each targeted server. Then, you execute the deploy and configure script with the appropriate properties file as a parameter.

Oracle recommends that you use the scripts described in this section to deploy your application rather than using ojdeploy. The best practice is to use the Ant task provided by JDeveloper to build your application, and then to use the scripts described in this section to deploy and configure the application. The ojdeploy command-line utility allows you to deploy your application from JDeveloper without starting the JDeveloper IDE.

Note: The propagation tool, described in Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production," is primarily used by site administrators to push approved site structure changes to production without incurring any downtime.

Tip: The build and deploy script isolates the information that is variable between environments, like the server names, ports, content management connections, database connections, and so on. Most aspects of the Framework application do not need to change when you redeploy it to another server. For example, the name of the application doesn't change.

10.11.2 Using the Deploy and Configure Script

The deploy and configure script described in this section is primarily used when the production application is mostly read only and live. In this scenario, content

contributors make active changes to the staged application and receive content approval before that content is moved to production. Oracle recommends that you use automatic content replication so that content is moved from stage to production after appropriate workflow approvals. See Section 10.14, "Propagating Content From Oracle WebCenter Content."

Security changes are not affected by the deploy and configure scripts. To push site structure changes to production, administrators can use the propagation tool, as described in Section 10.13, "Using the Propagation Tool to Propagate From Staging to Production." See also Section 10.15, "Managing Security Through the Life Cycle."

To deploy and configure your application to a target environment:

1. In a terminal window, go to the directory that contains the deploy and configure scripts. These scripts are called: `create_profile.csh` and `deploy_and_config.csh`. These files reside in `WEBCENTER_HOME/webcenter/scripts/stage2prod`, where `WEBCENTER_HOME` is the directory where WebCenter is installed.

   Note: The deploy and configure scripts in `stage2prod` are samples only. You are free to develop your scripts in a different location (after copying the sample and making changes to it for your deployed environment).

2. First, you need to provide some target environment-specific information in the `setup.properties` file, like the target server URL, user name, and password. Simply open the file `setup.properties` and add the appropriate values for the target environment. A sample file is shown in Example 10–1.

   Note: For WebLogic Sever deployments, you can ignore properties listed under WebSphere Server properties. See also, "Using the Deploy and Configure Script for Applications Deployed on WebSphere" in the *Oracle Fusion Middleware Third-Party Application Server Guide*.

Example 10–1 Sample setup.properties File

```
# Adminserver
admin.jmx.url=t3\://hostname\:7001
admin.user=weblogic
admin.password=welcome1

# Application
webcenter.app.name=webapp
webcenter.app.server=WC_CustomPortal
webcenter.app.version=V2.0
```

3. Update the `create_profile.csh` and `deploy_and_config.csh` to reflect the deployed environment.

   ```
   setenv WC_HOME <webcenter_home>
   setenv SCRIPTS_DIR <scripts_home>
   ```

   `WC_HOME` is the WebCenter Home and `SCRIPTS_DIR` is where the scripts are located. By default, the scripts are here: `$WC_HOME/webcenter/scripts/stage2prod`. If you copied the scripts to another location, then set `SCRIPTS_DIR` to that location.

4. Run the `create_profile` script. The input to this script is the `setup.properites` file. For example, in a Linux environment, enter:

   ```
   ./create_profile.sh
   ```

   This script examines your application environment and produces an output properties file called `profile.properties`.

5. If you wish, rename the output file, `profile.properties` to a name that reflects the target environment. For example, if the target environment is your stage environment, you might call the file output file `wstage.properties`.

   The `profile.properties` file specifies all the configuration information needed to run the portal on the target environment. For example, it includes settings for content management repository, omni portlet, WSRP producers, Personalization for WebCenter Portal and others. Example 10–2 shows a sample `profile.properties` file.

Example 10–2 Sample profile.properties File

```
webcenter.wcps.app.name=wcps-services
webcenter.wcps.app.server=WC_Utilities
admin.jmx.url=t3\://hostname\:7001
doclib.Content.cis.socket.host=hostname
app.mds.jndi=jdbc/mds/SpacesDS
webcenter.app.archive=/net/hostname/scratch/webapp.ear
doclib.Content.cis.socket.port=9444
webcenter.wcps.archive=/net/hostname/scratch/wcps.mar
webcenter.app.name=webapp
admin.user=weblogic
app.mds.repository=mds-SpacesDS
app.mds.partition=wcps-services
webcenter.app.version=V2.0
web.OmniPortlet.url=http\://hostname\:7101/portalTools/omniPortlet/providers/omniP
ortlet
app.restart=false
webcenter.app.server=WC_CustomPortal
```

> **Note:** All properties in the `profile.properties` must have a value. If a property is not needed, delete it or comment it out rather than leave the value empty.

6. Run `create_profile` to create a properties file for each of your target environments. For example, you might create one each for your test, stage, and production environments.

7. Run the `deploy_and_confg` script. The input to this script is the `profile.properties` file (or whatever you renamed the file). For example, in a Linux environment, might enter:

   ```
   ./deploy_and_config.csh
   ```

The `deploy_and_config` script takes one of two "modes" as input. These modes are `deploy_config` and `p13n_metadata`. For example:

```
./deploy_and_config.csh p13n_metadata
```

The `deploy_config` mode is the default mode if no input is passed to `deploy_and_config.csh`. The `deploy_config` mode does the deployment and configuration tasks. If you only need to update the update of personalization metadata, you can override the default behavior by passing in `p13n_metadata` as the input to the script.

This script deploys and configures the Framework application to run on the target environment.

10.11.3 Managing Post-Deployment Changes

All WebCenter Portal configuration changes performed after the application is deployed are stored as customizations in MDS. For example, if your application had some connections defined in `connections.xml`, and you then deployed the application on another managed server, and performed configurations using Oracle Enterprise Manager Fusion Middleware Control Console or WLST, then these changes are stored in MDS as customizations. Subsequently, if you deploy a newer EAR file, the connection changes you performed earlier would still be in effect, and will override the connection definition within the EAR file. The configuration changes performed with the Fusion Middleware Control Console or WLST persist after a redeployment. See also "Oracle WebCenter Portal Administration Tools" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

> **Note:** Portlet Producer application customizations are also stored in MDS. When you deploy a portlet producer, the customizations are uploaded to the currently configured producer connection. Oracle recommends that you use the deploy and configure scripts to manage the producer connections (rather than reconfiguring them after deployment and then redeploying the application). The deploy and configure scripts ensure that the producer customizations are redeployed correctly whenever the connection information changes.

10.12 Building and Deploying Personalization for WebCenter Portal Files

This section explains how to build and deploy Personalization for WebCenter Portal files.

> **Note:** For information on integrating and configuring WebCenter Personalization, see "Personalization for WebCenter Portal Prerequisites" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

- Section 10.12.1, "Building the WCPS MAR File"
- Section 10.12.2, "Deploying WCPS Files to a Server"
- Section 10.12.3, "Deploying WCPS Files to an Archive File"
- Section 10.12.4, "Importing and Exporting MAR Files"

10.12.1 Building the WCPS MAR File

WebCenter provides an Ant task for building the WCPS MAR file. Oracle recommends that you add this task to your nightly build script if you are using WCPS.

1. Create an Ant build script for your Framework application. For details, see Section 10.7, "Configuring a Nightly Build Script."
2. Open the `build.xml` file in an editor.
3. Add an Ant target for the `build-mar` task to the `build.xml` file, as shown in Example 10–3.

> **Note:** The Ant task supports nested `<fileset>` elements to specify the locations under which to scan for scenario and namespace XML files. This element supports all the standard Ant fileset semantics. Table 10–3 illustrates the use of `<fileset>`.

Example 10–3 Example deploy-wcps-mar Target Element

```
<target name="build-wcps-mar">
<taskdef name="build-mar"
        classname="oracle.wcps.tools.ant.BuildMARTask"
        uri="oracle:webcenter:wcps"> <classpath> <pathelement
path="${oracle.jdeveloper.home}/jdev/extensions/oracle.wcps.tools.common/oracle.wcps.tools.ant.jar"
/>
</classpath>
</taskdef>
<basename property="application.name"
         file="${oracle.jdeveloper.workspace.dir}"/>
<wcps:build-mar xmlns:wcps="oracle:webcenter:wcps"
               destFile="deploy/${application.name}-wcps.mar"
               appName="${application.name}"
               scenarionamespace="SomeThingElse"
               failonfilescanerror="false" failonempty="true"> <fileset
dir="${oracle.jdeveloper.workspace.dir}">
<include name="Portal/**/*.xml"/>
<exclude name="Portal/public_html/**"/>
</fileset>
</wcps:build-mar>
</target>
```

4. Add the task to the `depends` attribute on the `all` target. For example:

   ```
   <target name="all" depends="deploy,build-wcps-mar"/>
   ```

5. Save the `build.xml` file.

This `build-mar` Ant task supports the attributes listed in Table 10–3.

Table 10–3 Attributes Supported by the Ant Task

Attribute	Required	Description
destfile	Yes	The location of the `.mar` file to write. It will be overwritten if it exists.
appname	Yes	The application name. Typically, the base name of the `.jws` file.
scenarionamespace	No	The scenario namespace. Defaults to the application name, same (just as JDeveloper does).
failonfilescanerror	No	(Boolean) Specifies whether to have the task fail the build if it cannot scan a specified file. Defaults to true.
fileonempty	No	(Boolean) Specifies whether to have the task fail the build if no WCPS files are found. Defaults to `false`.

Note that if you do not set `failonfilescanerror="false"`, you might see errors similar to this one:

```
BUILD FAILED:
Unable to read
C:\opt\jdev-applications\Application1\Model\classes\.data\00000000\00000000.jdb
```

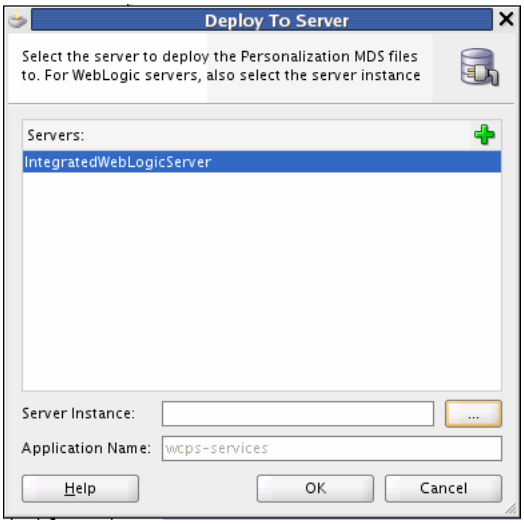
Either narrow your file sets (for example, use `*.xml`) or specify `failonfilescanerror="false"`, in which case it will just display a warning.

10.12.2 Deploying WCPS Files to a Server

To deploy WCPS files to a server:

1. In JDeveloper, open your portal project. If you want to use WLST commands to perform this deployment, see Section 10.12.4, "Importing and Exporting MAR Files."
2. From the Application menu, select **Deploy Personalization Files** and then select **To Server.**
3. In the Deploy to Server dialog, select the server to which you want to deploy the service. See Figure 10–3.

Figure 10–3 The Deploy To Server Dialog



Note: Optionally, click the **Add** button to add a server to the list.

4. Click the "**...**" button to select a specific server instance. For example, the running DefaultServer.
5. Click **OK** when you are finished.
6. Check, the Message - Log window for details on the deployment. The log will indicate if the deployment of the metadata files was completed successfully.

10.12.3 Deploying WCPS Files to an Archive File

You can deploy your WCSP files to a metadata archive (MAR) file on the file system. To deploy to a MAR file:

1. In JDeveloper, open your portal project.
2. From the Application menu, select **Deploy Personalization Files** and then select **To Filesystem.**
3. In the Deploy to Filesystem dialog, select **Create Metadata Archive** and specify a location and name for the MAR file.

The MAR file can then be updated to the server using command line utilities described in Section 10.12.4, "Importing and Exporting MAR Files."

10.12.4 Importing and Exporting MAR Files

To update a WCPS server with new files from a Metadata Archive (MAR) file, use the following WLST commands to connect to the server and import the metadata.

```
wlst:> connect('<username>', '<password>', 't3://admin-server-hostname:7001')
wlst:> importMetadata(application='wcps-services', server='WC_Utilities',
fromLocation='<mar file>', remote='true')
```

To export the WCPS from the server into a MAR file on the filesystem, use these WLST commands to connect to the server and export the metadata:

```
wlst:> connect('<username>', '<password>', 't3://admin-server-hostname:7001')
wlst:> exportMetadata(application='wcps-services', server='WC_Utilities',
toLocation='<mar file>', remote='true')
```

For more information on MDS WLST commands, refer to the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference* on the Oracle Technology Network.

If you're running on the IntegratedWebLogicServer in a development environment, the connect URL is `t3://localhost:7101` and the server is `DefaultServer`.

10.13 Using the Propagation Tool to Propagate From Staging to Production

The WebCenter Portal Administration Console includes a propagation tool for moving portal metadata from a staging to a production server. Site administrators use this tool occasionally to push approved site structure changes to the production server without incurring any downtime.

> **Note:** The Propagation tool supports propagating portal metadata and Personalization for WebCenter Portal files. For more information, see Section 10.13.4, "Propagating WCPS Files."

> **Note:** WebCenter provides WLST commands for importing and exporting portlet preferences. Use these commands to replicate portlet preference data between the stage and production servers. For more information, see Section 10.16, "Migrating Portlet Preferences."

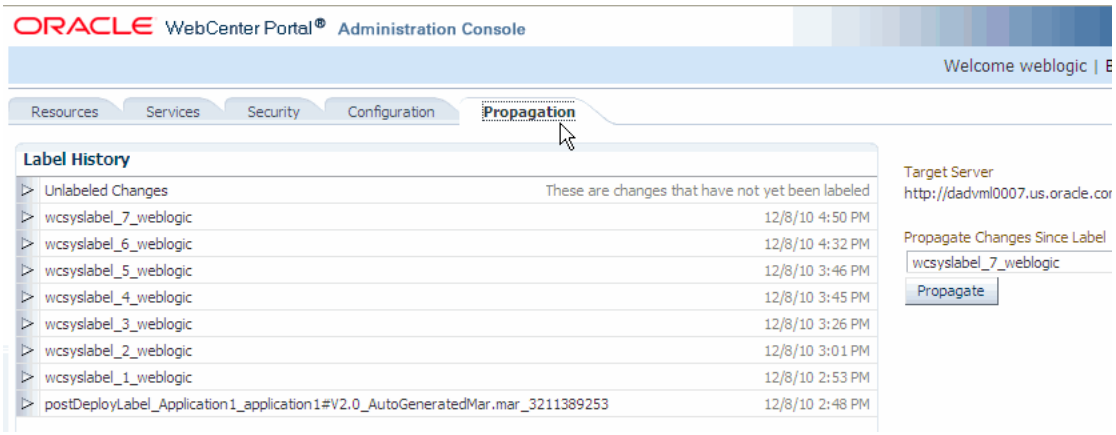
- Section 10.13.1, "Introduction to the Propagation Tool"
- Section 10.13.2, "Configuring the Propagation Tool"
- Section 10.13.3, "Propagating Portal Metadata"
- Section 10.13.4, "Propagating WCPS Files"

10.13.1 Introduction to the Propagation Tool

WebCenter provides a propagation tool to move a portal metadata from staging to production. In practice, the staging environment and the production remain identical until changes are made to the staged portal. When the changes are tested and approved, an administrator uses the propagation tool to "push" the changes to the production server. This transfer does not require the production server to be restarted.

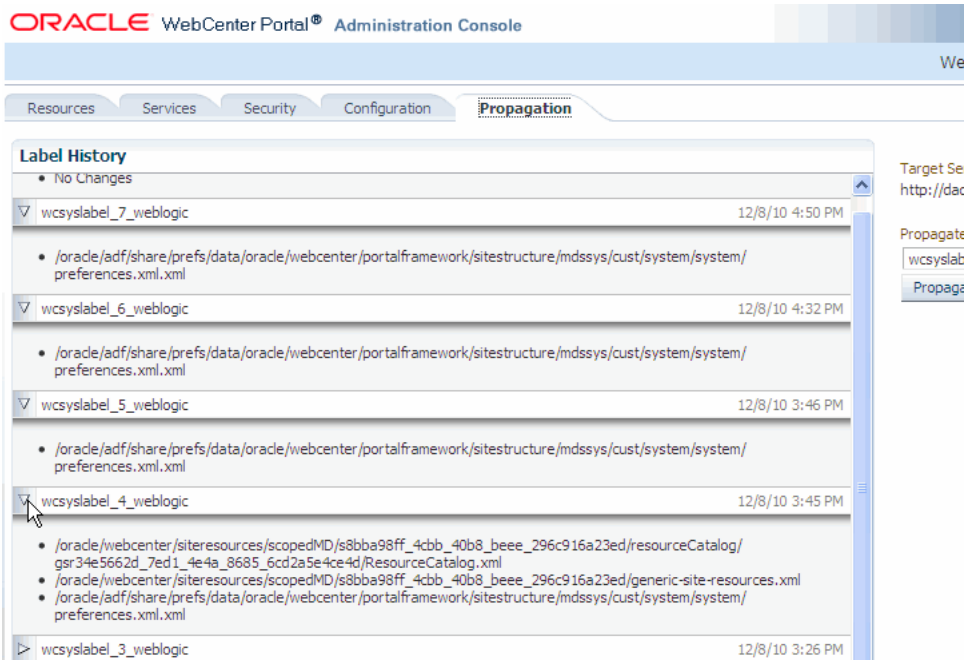
When properly configured, the Propagation tool shows up as a tab in the WebCenter Portal Administration Console, as shown in Figure 10–4.

Figure 10–4 The Propagation Tab



The Label History part of the Propagation tab lists the labels that were created for each propagation, labels created outside of the Propagation tab, such as labels created during deployment and labels created using WLST. Each label includes a list of files that change since the last propagation was performed. To view the contents of a label, click the arrow button to the left of the label name, as shown in Figure 10–5.

Figure 10–5 The Propagation Tab with Labels Opened



10.13.2 Configuring the Propagation Tool

> **Note:** If you are using the Propagation tool and if any of the MDS document file names contain multi-byte NLS characters, the stage machine should be configured for that NLS.

The Propagation tab only shows up in the WebCenter Portal Administration Console if the portal deployed to the staging server has a URL Connection configured. This URL connection must point to the production server and it must be named `ProductionURLConnection`.

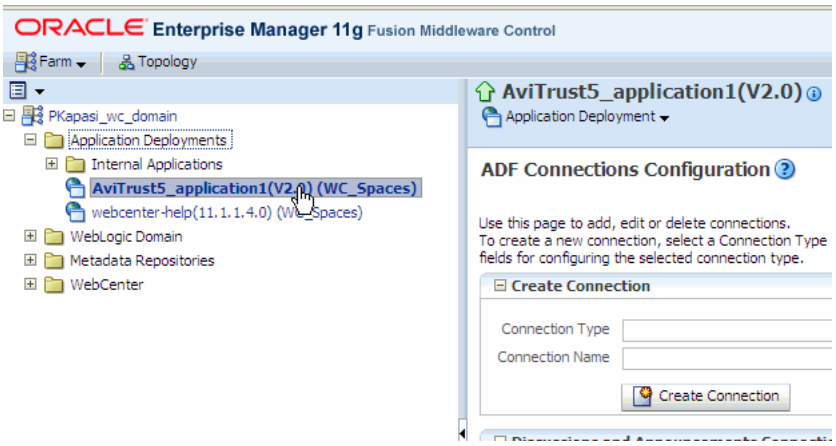
You have two choices for configuring the URL Connection: you can use the Fusion Middleware Control Console or Oracle WebLogic Scripting Tool (WLST). This section explains both techniques.

10.13.2.1 Configuring the URL Connection With Oracle Enterprise Manager Fusion Middleware Control Console

To configure the URL Connection with the Fusion Middleware Control Console:

1. In a browser, open the Fusion Middleware Control Console for your staging server domain. For details on starting the Fusion Middleware Control Console see *Oracle Enterprise Manager Administrator's Guide*.
2. In the navigation tree, open the Application Deployments folder and click on the application. See Figure 10–6.
3. In the application pane on the right, open the **Application Deployment** menu, then select **Configure ADF Connections** from the **ADF** submenu. The ADF Connections Configuration panel opens, as shown in Figure 10–6.

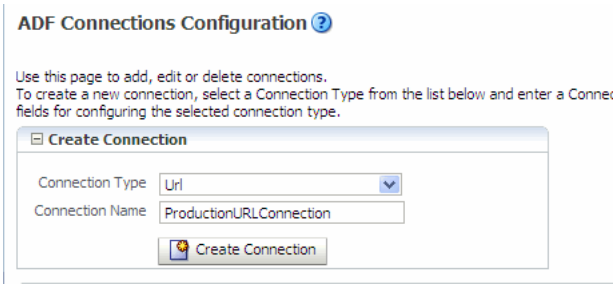
Figure 10–6 Selecting the Application in the Fusion Middleware Control Console



4. In the Create Connection box, select the URL connection type and enter `ProductionURLConnection` as the Connection Name. See Figure 10–7.

> **Note:** You must name the connection `ProductionURLConnection`. The name is case sensitive.

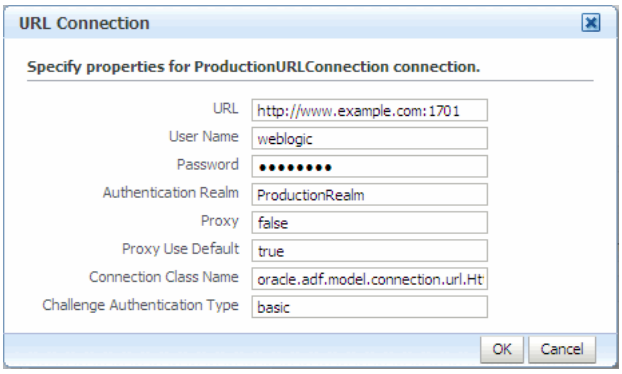
Figure 10–7 Create Connection



5. In the URL Connections box, click **Edit**.
6. Complete the URL Connection dialog box as follows.
   - **URL** – Enter the URL (including hostname and port number) of the production server.
   - **Username** – Enter a username for the connection.
   - **Password** – Enter a password for the connection.
   - **Authentication Realm** – You must enter a value for the Authentication Realm. Do not leave this field blank. You can enter any arbitrary string, such as `ProductionRealm`.

     You can accept the default values for the remaining fields. See Figure 10–8.

Figure 10–8 URL Connection Dialog



7. Click **OK** after you complete the URL Connection dialog.
8. Click **Apply** in the upper-right corner of the Fusion Middleware Control Console window.

10.13.2.2 Configuring the URL Connection with WLST

You can use a Oracle WebLogic Scripting Tool (WLST) command to create the URL Connection. Note that the parameters to the command are equivalent to the values entered in the URL Connection dialog in the Fusion Middleware Control Console, described in the previous section, Section 10.13.2.1, "Configuring the URL Connection With Oracle Enterprise Manager Fusion Middleware Control Console."

The following is an example of the WLST command with sample values specified for each parameter. Replace these values with values that pertain to your production server:

```
adf_createHttpURLConnection(appName='portalApp', name='ProductionURLConnection',
url='http://production_adminserver_host:7001', user='weblogic',
password='password_for_weblogic', realm='ProductionRealm'
```

For information on WLST, see the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference* on the Oracle Technology Network.

10.13.3 Propagating Portal Metadata

Use the propagation tool when you want to move changes made on the staging portal to the production portal.

1. Be sure the propagation tool is configured, as described in Section 10.13.2, "Configuring the Propagation Tool." To use the propagation tool, you must have Administrator's privileges.
2. Open the WebCenter Portal Administration Console. By default, the administration console is located at this URL:

   `http://<server>:<port>/<context_root>/admin`

For more information on using the WebCenter Portal Administration Console, see "Using the WebCenter Portal Administration Console" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

3. Select the Propagation tab.

> **Note:** If you don't see the Propagation tab it could be because you don't have administrator's privileges or the propagation tool was not configured properly as explained in Section 10.13.2, "Configuring the Propagation Tool."

4. Click **Propagate** to transfer the files to the production server.

10.13.4 Propagating WCPS Files

The propagation tool supports moving Personalization for WebCenter Portal files from staging to production. The propagation tool automatically exports metadata from the application's MDS repository on the source server, then imports the metadata to the application's MDS repository on the target server.

The ability for the propagation tool to move WCPS files depends on the following conditions:

- The Framework application's MDS repository must point to the same database as the `mds-wcpsDS` datasource. This is the default configuration. See, for example, Figure 10–2 for a general illustration. In general, propagation will work correctly if you do not change the default database configuration.
- On both the source and destination servers, the application's MDS store must use the same partition name as the MDS store for `wcps-services`. In a standard portal configuration, this partition is `webcenter-portal`.

Note: Typically, there is a one to two minute delay between when the propagation tool is initialized and when the WCPS REST services are notified of the MDS changes. This delay does not occur when you use the WLST commands to propagate WCPS files, as explained in Section 10.12.4, "Importing and Exporting MAR Files."

10.14 Propagating Content From Oracle WebCenter Content

Oracle recommends that you use either manual or automated replication to move content from one environment to another. Another option is to manually propagate content; however, this option is only recommended if the source and target servers are unable to communicate with each other.

For detailed information on the tools and options for propagating Oracle WebCenter Content content, see "Managing System Migration and Archiving" in the *Oracle WebCenter Content System Administrator's Guide for Content Server*. That chapter explains how to propagate Oracle WebCenter Content content by exporting and importing archive files. In addition, the chapter explains how to set up and use replication. Replication can automate the export, import, and transfer functions. For example, you can use replication to automatically export from one content server instance, transfer the archive to another computer, and import to another content server instance.

10.15 Managing Security Through the Life Cycle

This section discusses techniques for migrating security policies and credentials from one environment to another. Migration can be either automatic or manual. Automatic migration is typically used when the application is first deployed. Manual migration is used on redeployment.

10.15.1 Migrating Security Policies for First-Time Deployment

When `jazn-data.xml` with application policies is packaged in the EAR file, Oracle Platform Security Services (OPSS) performs policy migration based on the application configuration settings in `weblogic-application.xml`. Example 10–4 shows the recommended settings in `weblogic-application.xml` to achieve automatic policy migration for first-time deployment.

Example 10–4 Settings for Automatic Policy Migration (weblogic-application.xml)

```
<wls:application-param>
    <wls:param-name>jps.policystore.migration</wls:param-name>
    <wls:param-value>MERGE</wls:param-value>
</wls:application-param>

<wls:application-param>
    <wls:param-name>jps.policystore.removal</wls:param-name>
    <wls:param-value>OFF</wls:param-value>
</wls:application-param>

<wls:listener>

<wls:listener-class>oracle.security.jps.wls.listeners.JpsApplicationLifecycleListe
ner</wls:listener-class>
</wls:listener>
```

JpsApplicationLifecycleListener must be specified in `weblogic-application.xml` to enable policy and credential migration. Oracle recommends that `jps.policystore.migration` should be set to `MERGE`. With this configuration, polices packaged in `jazn-data.xml` will be always migrated during fist time deployment. Oracle also recommends that `jps.policystore.removal` should be set to `OFF` so that application policies are not deleted from the policy store when the application is undeployed.

10.15.2 Migrating Security Policies On Redeployment

On redeployment, it is important to avoid overwriting policy changes made on the production system. When `jps.policystore.migration` parameter is set to `MERGE`, all new policies from the stage environment will be merged with production policies. With this setting, any changes made to existing policies, such as changing permissions for a role, deleting a role, or removing membership of a role, will not be migrated during redeployment, because such migration would cause a conflict.

If you want to overwrite the production policies with the staging environment security policies packaged in `jazn-data.xml`, use the `migrateSecurityStore` WLST command. See *Oracle Fusion Middleware Application Security Guide* for details on `migrateSecurityStore` command.

10.15.3 Migrating Credentials for First-Time Deployment

Application credential migration support is similar to application policy migration. JpsApplicationLifecycleListener supports credential migration in application deployment and redeployment with extra security enforcements.

With application credentials in cwallet.sso and packaged in an EAR file in the `META-INF/` directory, OPSS will perform credential migration to system credential store based on the configuration settings in `weblogic-application.xml`.

Example 10–4 shows the recommended settings in `weblogic-application.xml` to achieve automatic credential migration for first-time deployment.

Example 10–5 Settings for Automatic Credential Migration (weblogic-application.xml)

```
<wls:application-param>
    <wls:param-name>jps.credstore.migration</wls:param-name>
    <wls:param-value> MERGE </wls:param-value>
</wls:application-param>
```

Oracle recommends that `jps.credstore.migration` should be set to `MERGE` so that at deployment and redeployment time all new credentials will be migrated. This setting will only migrate new credential key/value from `cwallet.sso`. Modification done to existing credential keys will not be migrated.

10.15.4 Migrating Credentials On Redeployment

If credentials are modified on stage and need to be moved to production, use the command `migrateSecurityStore` to migrate modified credentials. See *Oracle Fusion Middleware Application Security Guide* for details on `migrateSecurityStore` command.

10.16 Migrating Portlet Preferences

WebCenter provides WLST commands for importing and exporting portlet preferences. Exported preferences (customizations and personalizations) can be imported into a target application – for example, exported from the staging application and imported into the production application. For more information, see `exportPortletClientMetadata` and `importPortletClientMetadata` in "Oracle WebCenter Portal Custom WLST Commands" in the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*. Also ensure that export and import are enabled. See Chapter 61, "Coding Portlets."

See also Chapter 62.3, "Deploying a Portlet Producer Application to an Oracle WebLogic Managed Server Instance."

10.17 Rolling Back Production Site Changes

In rare instances, it might be necessary to roll back changes that are pushed to the production server from staging. The propagation model described in this chapter assumes that you are making a small number of changes to the staging instance and then pushing them to production using the propagation tool. If you ever need to roll back those changes, Oracle recommends that you manually undo the changes on the staging instance and push the modified staging instance back to production, after adequate validation and testing. By this process, you will have "rolled back" the original changes and the production system will be reverted to its original state.

11

Building a Navigation Model for Your Portal

This chapter describes how to create the navigation model in your WebCenter Portal: Framework application. Navigation models define the content, structure, and metadata of your portal navigation.

This chapter includes the following sections:

- Section 11.1, "Introduction to Navigation Models"
- Section 11.2, "Creating a Navigation Model"
- Section 11.3, "Adding Resources to a Navigation Model"
- Section 11.4, "Selecting the Default Navigation Model"
- Section 11.5, "Editing the Navigation Renderer"
- Section 11.6, "Setting the REDIRECT_OPTIONS Context Parameter"
- Section 11.7, "Enabling Runtime Administration of Navigation Models"
- Section 11.8, "Working with WebCenter Portal: Spaces Navigation Models"

For information about how to expose the navigation model in your portal, see Chapter 14, "Visualizing Your Portal Navigation."

11.1 Introduction to Navigation Models

Typically, portals provide information from various different sources. Information may be provided by pages, portlets, task flows, content repositories, and external pages. Users need a way to easily move through this information and quickly access the specific information that they need. You provide this access to information through your portal navigation.

The *navigation model* defines the content, structure, and metadata of the navigation. When you create the navigation model, you specify the items to include and the hierarchy of those items. Navigation models can include the following resources:

- Pages (individual pages and page hierarchies)
- Content (individual content items or the results of a content query)
- Other navigation models
- Portlets
- Task flows
- External applications
- External links

For more information, see Section 11.3, "Adding Resources to a Navigation Model."

You can also add resources that are dynamically generated. These are:

- Custom Resources
- Components
- Custom Folders
- Custom Content

For more information, see Section 11.3.7, "How to Extend Navigation Model Data."

When you create a new Framework application, a navigation model is created for you (`default-navigation-model.xml`). You can build on this seeded navigation model or create your own. You might even want to create different navigation models for different user types.

> **Note:** If you create your own navigation model, rather than extend the seeded navigation model, we recommend that you set your navigation model as the default navigation model. For more information, see Section 11.4, "Selecting the Default Navigation Model."

After you have created your navigation model, you can expose it on any page template in your portal either through the built in navigation task flows, or through your own navigation UI. For more information, see Chapter 14, "Visualizing Your Portal Navigation."

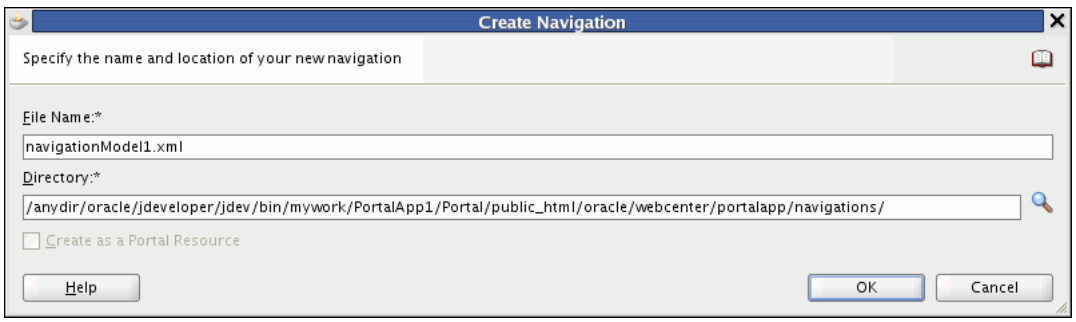
11.2 Creating a Navigation Model

If the default navigation model provided out of the box does not meet your requirements, you can create your own navigation model.

To create a navigation model:

1. In the Application Navigator, right-click the node where you want to create the navigation model (for example `/oracle/webcenter/portalapp/navigations`) and choose **New**.
2. In the New Gallery, expand **Web Tier**, select **Portal** and then **Navigation**, and click **OK**.
3. In the Create Navigation dialog (Figure 11–1), in the **File Name** field, enter a name for the XML file that represents the navigation, for example `myNavigation.xml`.

Figure 11–1 The Create Navigation Dialog



4. In the **Directory** field, enter the full directory path of the location under which to create the file.

   Note: In a Framework application, to expose the new navigation model in the runtime Resource Manager, you must create the navigation model under the *Application_Root*`/Portal/public_html/oracle/webcenter/portalapp` directory. By default WebCenter Portal includes a separate `navigations` directory.

5. Select **Create as a Portal Resource** to add the navigation model to the resources registry, allowing it to be managed at runtime through the Resources Manager.

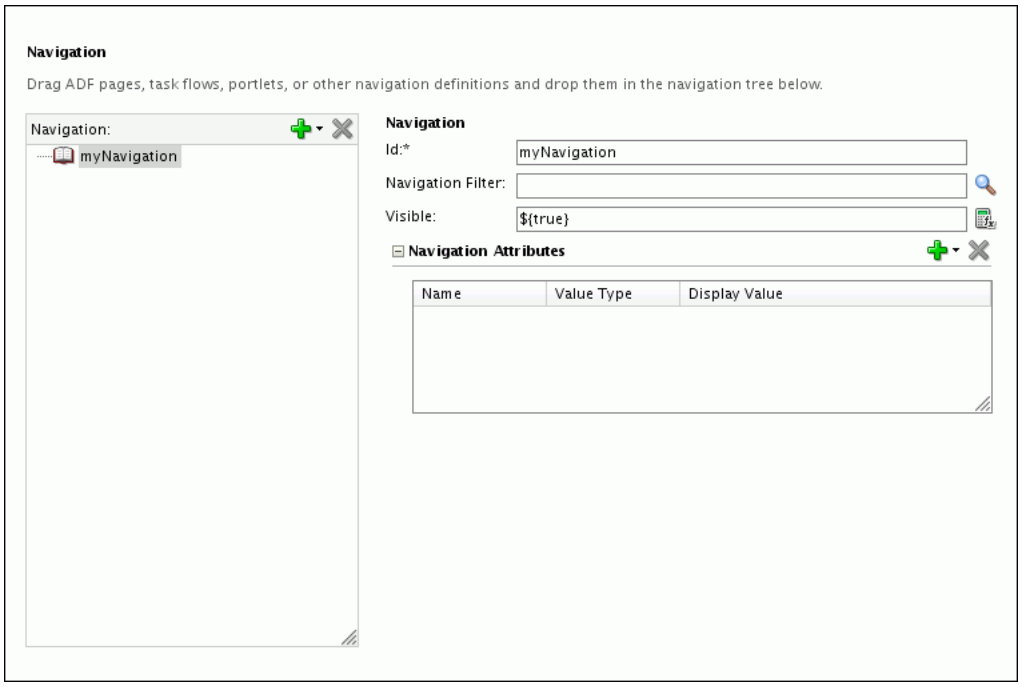
   Note: This option is available only if the directory you entered is the *Application_Root*`/Portal/public_html/oracle/webcenter/portalapp` directory or one of its subdirectories.

   Tip: If you do not select this option now, you can add the navigation model to the resources registry later by right clicking it and choosing **Create Portal Resource**.

6. Click **OK**.

   The new navigation model opens in Design view (Figure 11–2).

Figure 11–2 A Navigation Model in Design View



7. In the Design view for the navigation model, select the new navigation model in the Navigation column and define general properties as required:

- In the **Navigation Filter** field, enter the name of a Java class to use to filter out selected resources in the navigation model based on specific criteria. The Java class should implement the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter` interface.

  Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed when the navigation model is rendered on a page.

  Note: You can also conditionally hide individual resources in the model by specifying an EL expression in the **Visible** field for the resource.

- In the **Visible** field, specify whether the navigation model is available for use in the application. This field takes an EL value. The default is `#{true}`, which means the navigation model can be used by other users and is available in dropdown lists of navigation models.
- Use the Navigation Attributes section to set display options for the navigation model as desired. The options available are the same as those for individual resources. For more details about the different display options, see Section 11.3.9, "How to Set Display Options for a Resource.".

8. Save the navigation model definition file.

   Now that you have created the navigation, you can add resources to it. For more information, see Section 11.3, "Adding Resources to a Navigation Model."

11.3 Adding Resources to a Navigation Model

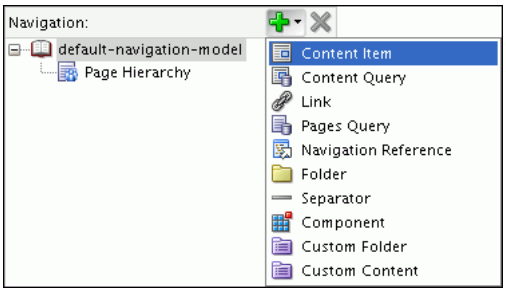
To define the content of your navigation model you can add resources to the default navigation model, `default-navigation-model.xml`. The default navigation model is created automatically when you create an application using WebCenter Portal's Framework application template.

You can also create your own custom navigation model and add resources to that. For more information, see Section 11.2, "Creating a Navigation Model."

You can add resources to a navigation model in two ways:

- Drag and drop resources from different areas in JDeveloper onto the navigation model in Design view.
- Use the options on the **Add new node** menu in the Design view for the navigation model (Figure 11–3).

Figure 11–3 The Add New Node Menu



In the Design view for the navigation model you can also modify and remove existing resources.

This section includes the following subsections:

- Section 11.3.1, "How to Drag and Drop a Resource onto a Navigation Model"
- Section 11.3.2, "How to Add a Content Item to a Navigation Model"
- Section 11.3.3, "How to Add Content Based on the Results of a Query to a Navigation Model"
- Section 11.3.4, "How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model"
- Section 11.3.5, "How to Add a Page Hierarchy to a Navigation Model"
- Section 11.3.6, "How to Embed Another Navigation Model Within a Navigation Model"
- Section 11.3.7, "How to Extend Navigation Model Data"
- Section 11.3.8, "How to Organize Resources in a Navigation Model"
- Section 11.3.9, "How to Set Display Options for a Resource"

11.3.1 How to Drag and Drop a Resource onto a Navigation Model

You can add the following resources to a navigation model by dragging and dropping them onto the desired location in the Design view for navigation model:

- **Content**—You can drag a content item from an Content Server connection in the Application Resources pane or Resource Palette onto a navigation model.
- **Pages**—When you drag a page from the Application Navigator onto a navigation model, you are prompted to choose whether to add just the page itself or to add a page query that includes the page its subpages.
- **Portlets**—You can drag a portlet from the Application Resources pane or Resource Palette onto a navigation model.
- **Task flows**—You can drag a task flow from the Application Resources pane or from the Resource Palette onto a navigation model.
- **External applications**—You can drag an external application from the Application Resources pane or Resource Palette onto a navigation model.
- **URLs**—You can drag a URL connection from the Application Resources pane or Resource Palette onto a navigation model.
- **Navigation models**—You can drag a navigation model from the Application Navigator and embed it in another navigation model.

11.3.2 How to Add a Content Item to a Navigation Model

In your navigation model, you can include links to individual content items within an existing content repository connection, or you can include a folder to provide access to all its contents.

You can also include a set of documents that match particular query criteria to a navigation model. For more information, see Section 11.3.3, "How to Add Content Based on the Results of a Query to a Navigation Model."

To add a content item:

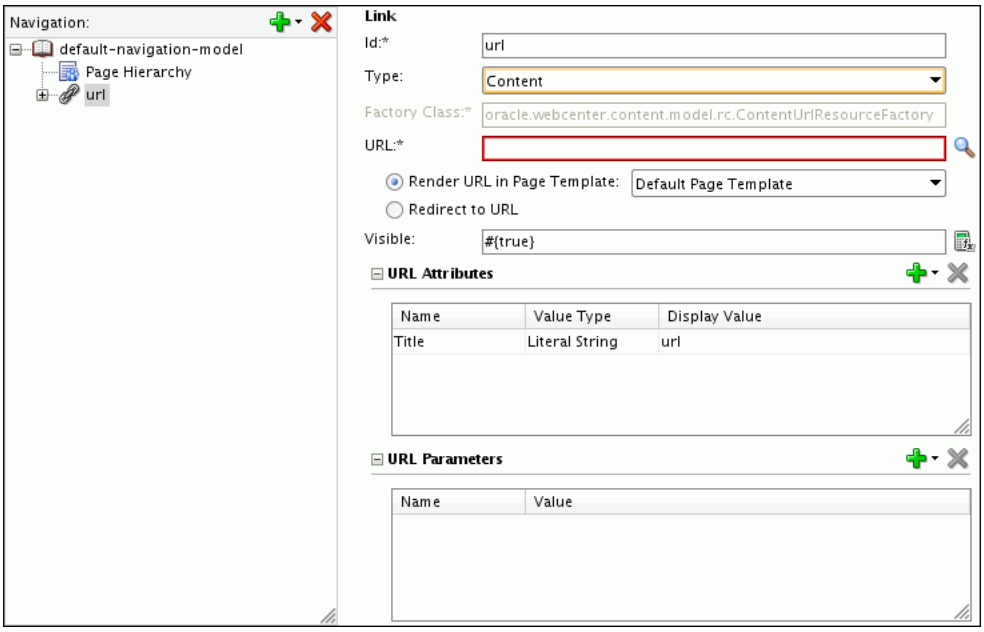
Tip: To add a content item to a navigation model you can also drag and drop it into the Design view for the navigation model.

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Content Item**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–4 Adding a Content Item to a Navigation Model



5. In the **URL** field, enter or browse for the URL to access the resource. If you enter the URL manually, use the following format:

   `content://Content_Connection_Id/Document_Id`

   Note: If the following is true, you must make sure that an appropriate connection is added to the application (in the Application Resources pane) to ensure that the resource is available at runtime:

   - You manually entered the URL
   - The connection exists in the Resource Palette

6. Select **Render URL in Page Template**, to render the resource in the current page, using the specified page template.

   Select **Redirect to URL**, to overwrite the current page with the resource when it is selected in the navigation.

7. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed when the navigation model is rendered on the page.

8. In the URL Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

9. If you want to use a Content Presenter display template to display the content item, in the URL Parameters section, click the **Add** icon, and choose **templateView**.

10. In the **Value** field for the **templateView** parameter, enter the display template view ID to use for rendering the content item, for example:

    ```
    oracle.webcenter.content.templates.default.detail
    ```

    Note: If you display a wiki page in a Framework application using a Content Presenter display template, by default any links within that wiki page are displayed in the Document Viewer. If you want to display wiki page links using Content Presenter, you must edit the `adf-config.xml` file. For more information, see Section 32.3.5, "Displaying Wiki Page Links Within Content Presenter."

11. Add further parameters, as required. For information about additional Content Presenter parameters, see Section 31.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates."

12. Save the navigation model definition file.

11.3.3 How to Add Content Based on the Results of a Query to a Navigation Model

You can add a content query to a navigation model to include all documents that match specific search criteria.

Note: To add a single file, or all the files within a particular folder, to a navigation model, see Section 11.3.4, "How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model."

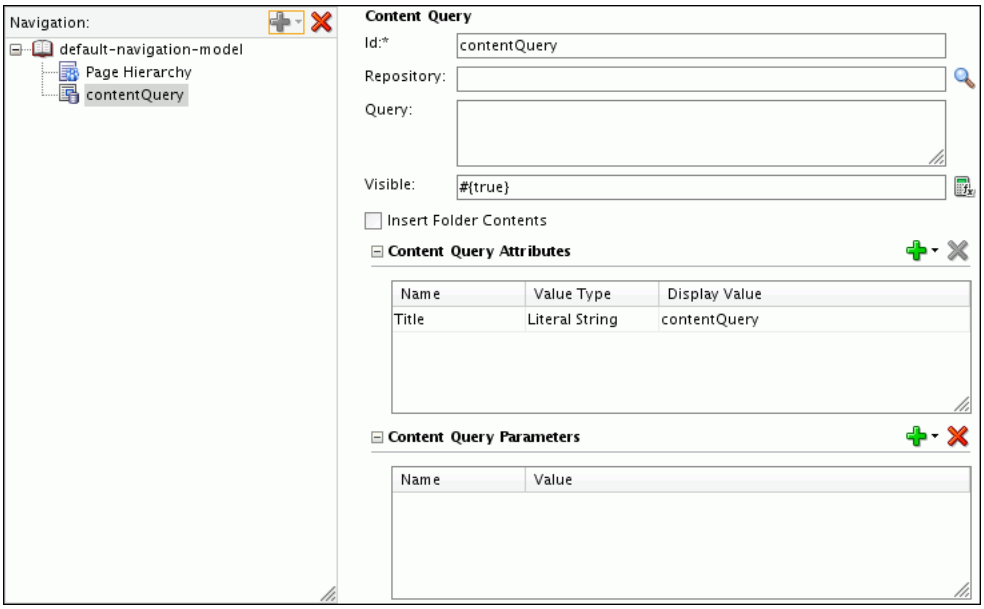
To add a content query:

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Content Query**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–5 Adding a Content Query to a Navigation Model



5. In the **Repository** field, enter or browse for the content repository whose content you want to search.

   Note: If the following is true, you must make sure that the appropriate connection is added to the application (in the Application Resources pane) to ensure that the resource is available at runtime:

   - You manually entered the repository name
   - The connection exists in the Resource Palette

6. In the **Query** field, enter the query criteria to identify the content to include in the navigation model, for example:

```
select * from cmis:document WHERE cmis:name LIKE 'Foo%'
```

For more information about how to format the query and for more examples, see *Oracle Fusion Middleware Content Management REST Service Developer's Guide*.

7. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

8. Select **Insert Folder Contents** to display the results of the query directly in the navigation, rather than displaying them under a folder.
9. In the Content Query Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."
10. If you want to use a Content Presenter display template to display the results of the content query, in the URL Parameters section, click the **Add** icon, and choose **templateView**.
11. In the **Value** field for the **templateView** parameter, enter the display template view ID to use for rendering the results of the content query, for example:

    ```
    oracle.webcenter.content.templates.default.list.simple
    ```

    Note: If you display a wiki page using a Content Presenter display template, by default any links within that wiki page are displayed in the Document Viewer. If you want to display wiki page links using Content Presenter, you must edit the `adf-config.xml` file. For more information, see Section 32.3.5, "Displaying Wiki Page Links Within Content Presenter."

12. Add further parameters, as required. For information about additional Content Presenter parameters, see Section 31.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates."

    Note: If you include links to the same portlet twice in a navigation model, the first link specifying parameter values and the second not, any parameters set for the first link are retained and also used for the second. To ensure that parameters are not used for the second link, you must explicitly set the parameter values to null.

13. Save the navigation model definition file.

11.3.4 How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model

You can add links to the following types of resources to a navigation model:

- **External Link:** Hypertext link to any external web page.

  Note: If you use the External Link option to include a `mailto:` link in your navigation, your navigation UI must explicitly handle this type of link. For information, see Section 12.2.3, "What You Should Know About Creating Page Templates for WebCenter Portal Applications."

- **External Application:** Link to any registered external application.
- **Page:** Link to any JSPX page in the application.

  This option enables you to add a link to a single page to the navigation model. For information about how to include an entire page hierarchy, see Section 11.3.5, "How to Add a Page Hierarchy to a Navigation Model."

- **Task Flow:** Link to any task flow in the application, or to a WebCenter Portal task flow (available in the Resource Palette).

  Note: If you provide navigation directly to a task flow in this way, users cannot customize the task flow. If you want users to be able to customize the task flow, add it to a page and add that page to the navigation model.

- **Portlet:** Link to a portlet in any registered portlet producer.

  Note: If you provide navigation directly to a portlet in this way, users cannot customize the portlet. If you want users to be able to customize the portlet, add it to a page and add that page to the navigation model.

- **Other:** Link to a custom resource. For more information, see Section 11.3.7.1, "How to Add a Link to a Custom Resource."

To add a link to a page, portlet, task flow, external application, or web page:

Tip: To add a portlet, task flow, external application, or web page to a navigation model you can also drag and drop it into the Design view for the navigation model.

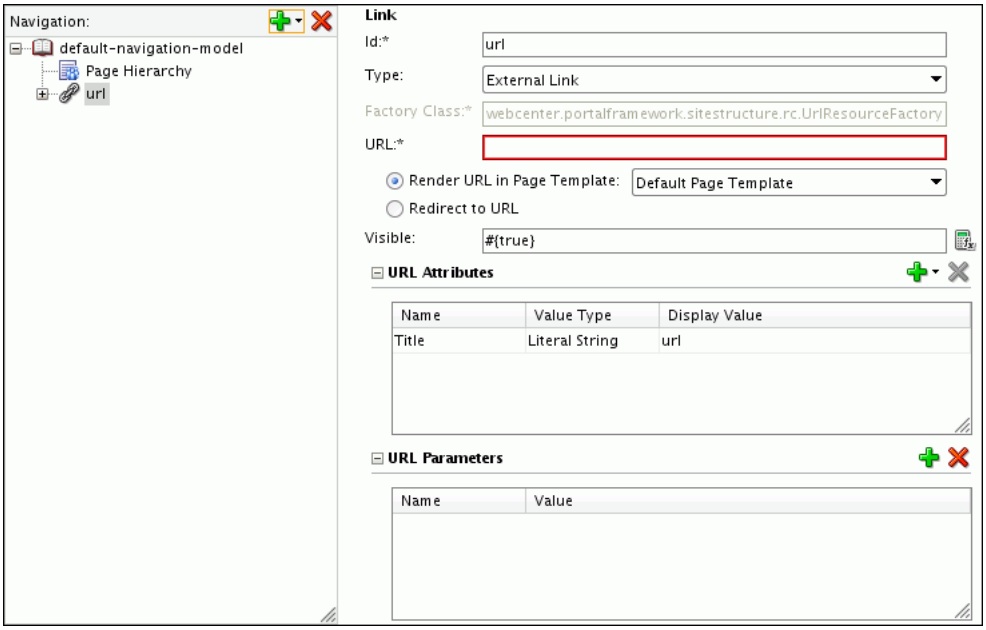
To add a link to a single page, drag the page into the Design view for the navigation model and choose **Page Link** from the popup menu.

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Link**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

> **Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–6 Adding a Link to a Navigation Model



5. From the **Type** dropdown list, select the type of resource that you want to add to the navigation model: **External Link**, **External Application**, **Page**, **Taskflow**, **Portlet**.
6. The **Factory Class** field is automatically populated with the appropriate factory class for the resource type and cannot be edited.

> **Note:** If you are adding a resource of type **Other**, the **Factory Class** field is editable and you must enter the factory class of the resource. For more information, see Section 11.3.7.1, "How to Add a Link to a Custom Resource."

7. In the **URL** field, enter or browse for the URL to access the resource. Table 11–1 provides the URL format to use for the different resource types.

Table 11–1 Format for URL Different Resource Types

Resource Type	Location in Application	URL Format
External Link	None	Absolute or relative path to web page **Note:** If you use the External Link option to include a `mailto:` link in your navigation, your navigation UI must explicitly handle this type of link. For information, see Section 12.2.3, "What You Should Know About Creating Page Templates for WebCenter Portal Applications."
External Application	External application connection in the Application Resources pane	`extapp://External_Application_Id`

Page	Application Navigator	`page://Path_to_Page/Page_Name`
		For example:
		`page://oracle/webcenter/portalapp/pages/myPage.jspx`

Task Flow from an ADF Library	Resource Palette	`taskflow://Path_to_TaskFlow/TaskFlow_Definition_File_Name#TaskFlow_Id`
Task Flow Within the Application	Application Navigator	`taskflow://Path_to_TaskFlow/TaskFlow_Definition_File_Name#TaskFlow_Id`
Portlet	Portlet producer connection in the Application Resources pane or in the Resource Palette	`portlet://Producer_Id/Portlet_Id`

Note: If the following is true, you must make sure that an appropriate connection is added to the application (in the Application Resources pane) to ensure that the resource is available at runtime:

- You manually entered the URL (without using drag and drop or the Choose a Resource dialog)
- The link **Type** is **External Application**, or **Portlet**
- The connection exists in the Resource Palette

8. Select **Render URL in Page Template**, to render the resource in the current page, using the specified page template.

   Select **Redirect to URL**, to overwrite the current page with the resource when it is selected in the navigation.

9. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed when the navigation model is rendered on the page.

10. In the URL Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."
11. In the URL Parameters section, enter values, as desired, for any parameters supported by the selected resource.
12. Save the navigation model definition file.

11.3.5 How to Add a Page Hierarchy to a Navigation Model

You can add a hierarchy of pages to the navigation model. All pages below the selected page in the hierarchy are included in the navigation model.

Although there is only a single page hierarchy within an application, in a navigation model you can use any node within the application page hierarchy as the starting point.

> **Note:** If you just want to include a single page in the navigation model, use the Link navigation type. For more information, see Section 11.3.4, "How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model."

To add a page hierarchy:

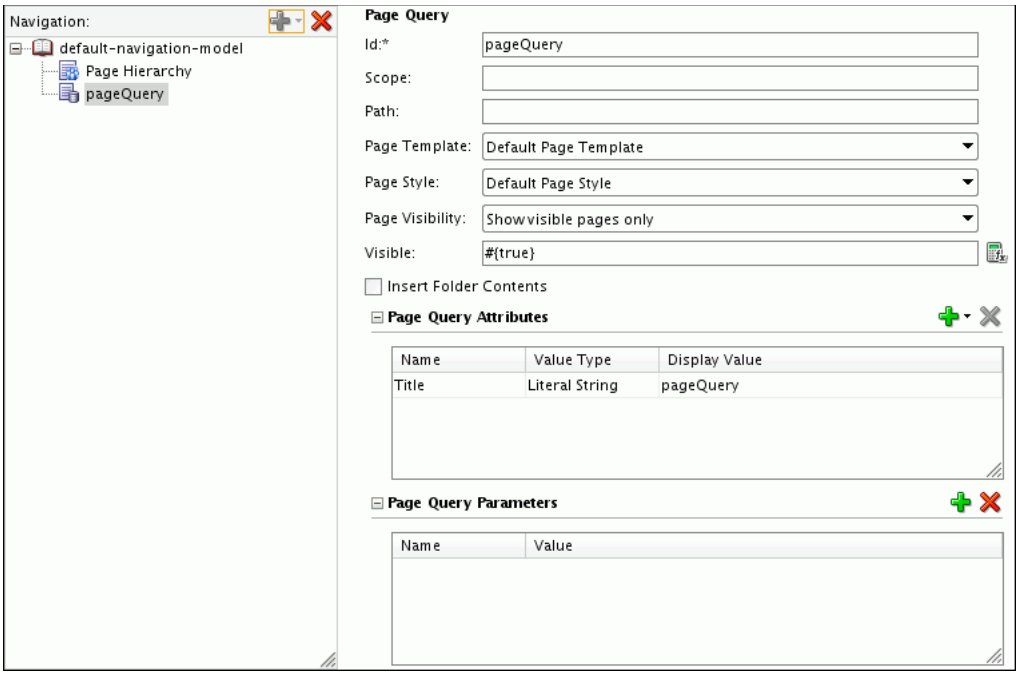
> **Tip:** To add a section of the page hierarchy to a navigation model, you can drag and drop a page (which has been added to the page hierarchy) from the Application Navigator into the Design view for the navigation model. Choose **Page Query** from the popup menu that appears.

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Page Query**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   > **Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–7 Adding a Page Query to a Navigation Model



5. The **Scope** field is not applicable to Framework applications (the scope is always the current application), so you can leave this field empty.

   Scope is used in WebCenter Portal: Spaces to determine whether to display all the pages in the Home space or a specific space if a particular page is not specified as the root of the page query.

6. In the **Path** field, enter the path and file name of the XML file that corresponds to the page within the page hierarchy that you want to use as the starting point of the page query. For example:

   ```
   /oracle/webcenter/portalapp/pagehierarchy/pages/pages.xml
   /oracle/webcenter/portalapp/pagehierarchy/pages/productsPages.xml
   ```

7. From the **Page Template** dropdown list, select the page template to use to display the resource when it is selected in the navigation.

8. From the **Page Style** dropdown list, select the style of page to include in the navigation model. For example, if you select **Wiki**, only pages that use the Wiki page style are included in the navigation..

9. From the **Page Visibility** dropdown list, select whether to include pages even if they have been flagged as hidden in the hierarchy.

10. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

> **Note:** Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

11. Select **Insert Folder Contents** to display the subpages in the hierarchy directly in the navigation rather than displaying them under a folder for the parent page.
12. In the Page Query Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."
13. In the Page Query Parameters section, enter values, as desired, for any parameters supported by the page template.
14. Save the navigation model definition file.

11.3.6 How to Embed Another Navigation Model Within a Navigation Model

You can nest navigation models within each other. For example, if you have a complex navigation model, it might be easier to break it down into several separate models and then nest them all within a single overarching navigation model. You can also use nested navigation models to delegate the development of navigation models to different users or to control access to different parts of the navigation.

To embed another navigation model within a navigation model:

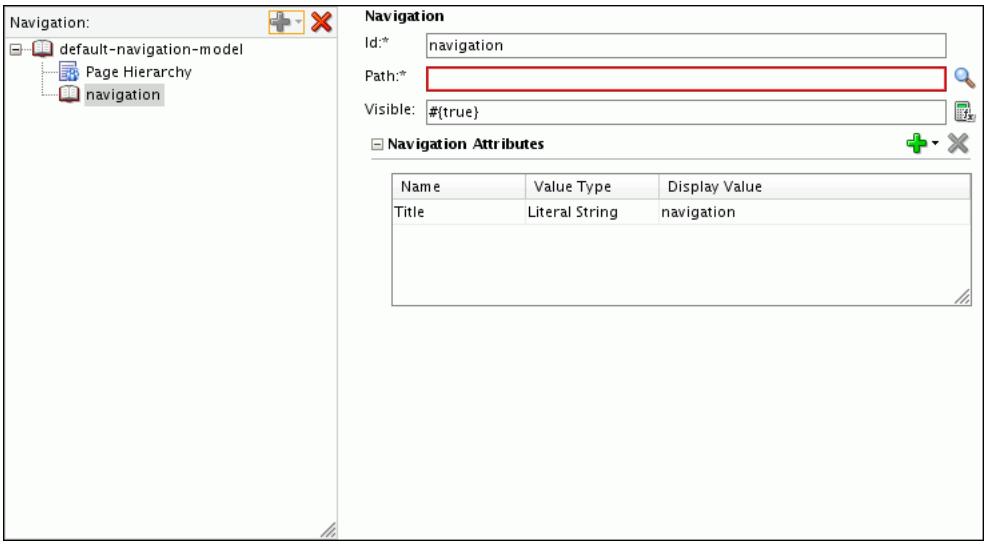
> **Tip:** To embed another navigation model within a navigation model you can also drag and drop the navigation model that you want to embed from the Application Navigator into the Design view for the navigation model.

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Navigation**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   > **Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–8 Embedding Another Navigation Model Within a Navigation Model



5. In the **Path** field, enter or browse for the location and file name of the navigation model definition file for the navigation model that you want to embed, for example:

   `/oracle/webcenter/portalapp/navigations/myNavigation2.xml`

6. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

7. In the Navigation Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

8. Save the navigation model definition file.

11.3.7 How to Extend Navigation Model Data

As well as adding known resources to your navigation model, you can also add resources to the model that are dynamically generated at runtime.

This section includes the following subsections:

- Section 11.3.7.1, "How to Add a Link to a Custom Resource"
- Section 11.3.7.2, "How to Add a Component to a Navigation Model"
- Section 11.3.7.3, "How to Add a Custom Folder to a Navigation Model"
- Section 11.3.7.4, "How to Add a Custom Content Provider to a Navigation Model"

11.3.7.1 How to Add a Link to a Custom Resource

You can add links to various WebCenter Portal-supported resources, such as pages, portlets, and task flows, by adding a link of the appropriate Type and identifying the URL of the resource. For more information, see Section 11.3.4, "How to Add a Page, Portlet, Task Flow, External Application, or Web Page to a Navigation Model."

If you want to link to a third party resource, you can do so by adding a link of type Other, and specifying the third-party `URLResourceFactory` implementation class.

To add a link to a custom resource:

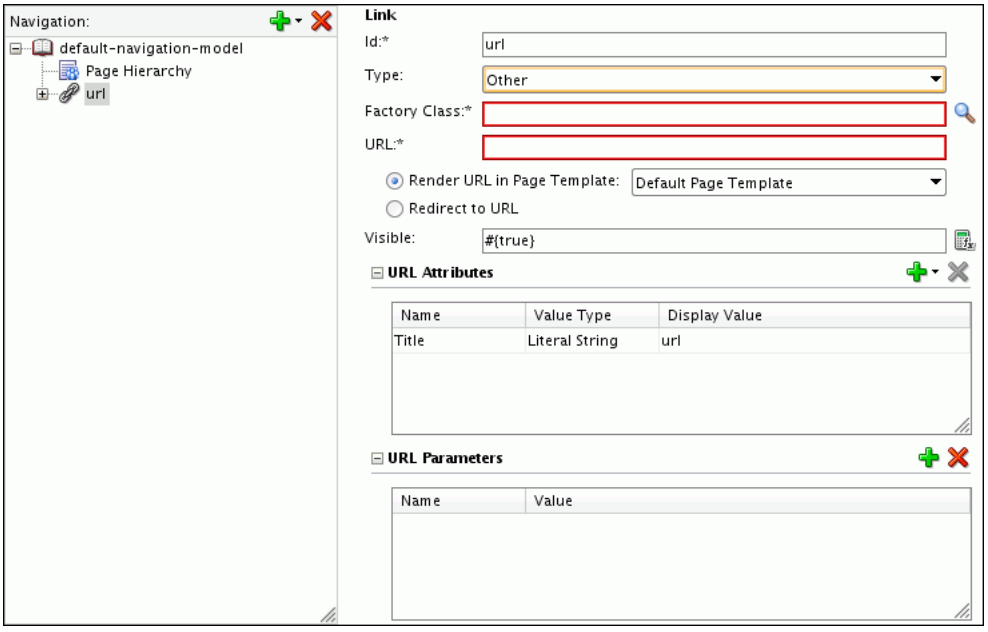
1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Link**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   > **Tip:** If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

5. From the **Type** dropdown list, select **Other**.

Figure 11–9 Adding a Custom Resource to a Navigation Model



6. In the **Factory Class** field, enter or browse for the `URLResourceFactory` implementation class that can resolve the URL for the custom resource.
7. In the **URL** field, enter the URL to access the resource.

8. Select **Render URL in Page Template**, to render the resource in the current page, using the specified page template.

   Select **Redirect to URL**, to overwrite the current page with the resource when it is selected in the navigation.

9. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

10. In the URL Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

11. In the URL Parameters section, enter values, as desired, for any parameters supported by the selected resource.

12. Save the navigation model definition file.

11.3.7.2 How to Add a Component to a Navigation Model

You can add a custom component to a navigation model. You can add a variety of components, such as simple ADF Faces components, compound objects containing two or more components, and JSF Verbatim tags that let you add arbitrary HTML content inside them. An example for arbitrary HTML that you can add to your catalog is a YouTube video.

To add a custom component:

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.

2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.

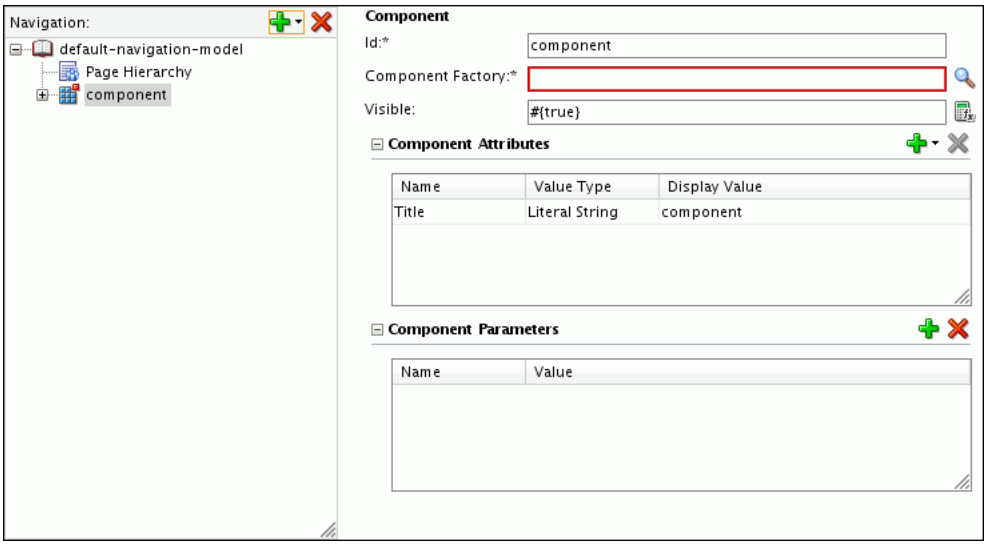
3. Click the **Add new node** icon and choose **Component**.

4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–10 Adding a Custom Component to a Navigation Model



5. In the **Component Factory** field, enter or browse for the Java class that implements the `oracle.adf.rc.component.ComponentFactory` interface and creates an instance of the component based on a set of parameters, for example, `oracle.adf.rc.component.XmlComponentFactory`.

   As you type the name, the field autocompletes with valid factory classes from the classpath.

   For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

6. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

7. In the Component Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

8. In the Component Parameters section, add the parameters that define the component.

   For example, if you are adding an XML component, click the **Add** icon in the Parameters section and select **xml**. In the new row that is displayed, specify the XML code for the component in the **Value** field.

   Note: The **xml** parameter is displayed only if you selected `oracle.adf.rc.component.XmlComponentFactory` as the component factory.

To ensure that the component works properly, you must specify the custom component ID to be #, and provide the namespace, as shown in the following example:

```
<cust:panelCustomizable id="#"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/>
```

The # value ensures that a unique ID is generated dynamically for the component.

9. Save the navigation model definition file.

11.3.7.3 How to Add a Custom Folder to a Navigation Model

You can add a custom folder to a navigation model that is configured to display its content dynamically at runtime. A custom folder is not added as a folder with resources; it contains only a reference to the factory class, which displays the resources dynamically.

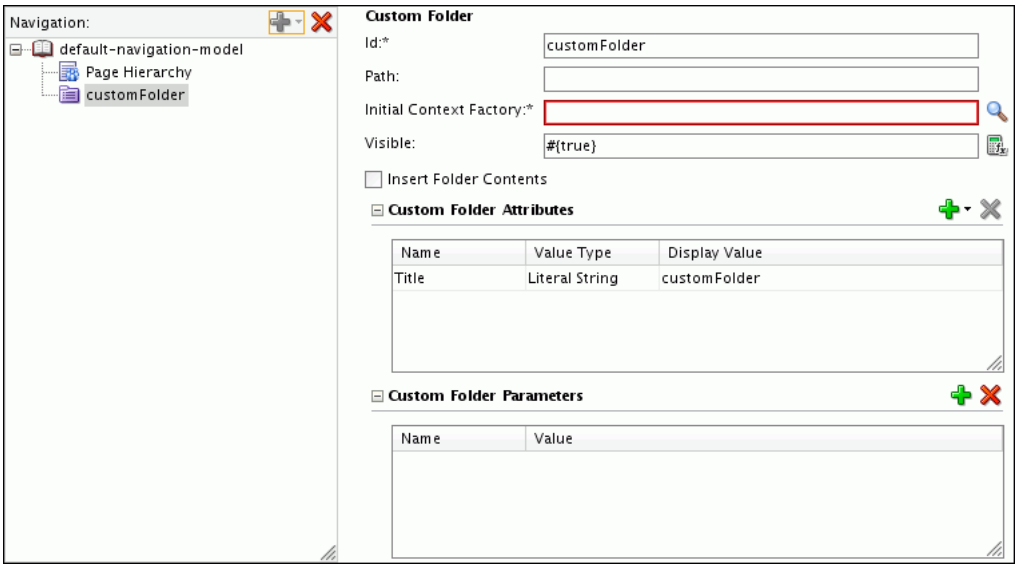
To add a custom folder:

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.
2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.
3. Click the **Add new node** icon and choose **Custom Folder**.
4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–11 Adding a Custom Folder to a Navigation Model



5. In the **Path** field, enter the MDS path for the custom folder.

6. In the **Initial Context Factory** field, enter or browse for the Java class that implements the `javax.naming.spi.InitialContextFactory` interface and generates the folder based on a set of parameters.

   For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

7. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

8. Select **Insert Folder Contents** to display the contents of the custom folder directly in the navigation rather than displaying them under the folder.

9. In the Custom Folder Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

10. In the Custom Folder Parameters section, add the parameters that were defined while implementing the factory class. You can bind these parameters to other artifacts in the application.

11. Save the navigation model definition file.

11.3.7.4 How to Add a Custom Content Provider to a Navigation Model

A custom content provider dynamically generates zero or more navigation items at runtime. It is very similar to a custom folder, except that, for custom folders, the folder is always displayed when the navigation model is rendered at runtime, even if it is empty. With custom content providers however, the folder is displayed at runtime only if it has content inside it.

To add custom content provider:

1. In the Application Navigator, right-click the navigation model to which you want to add the resource, and choose **Open**.

2. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the resource.

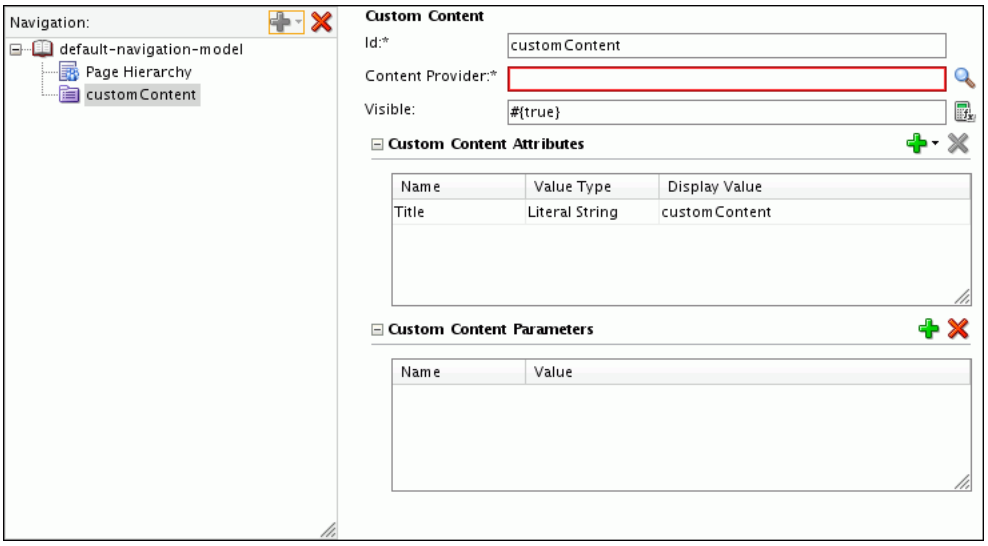
3. Click the **Add new node** icon and choose **Custom Content**.

4. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

   If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

   Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

Figure 11–12 Adding a Custom Content Provider to a Navigation Model



5. In the **Content Provider** field, enter or browse for the Java class that implements the `oracle.adf.rc.spi.plugin.catalog.CustomContentProviderV2` interface.

   For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

6. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.

   Note: Resources are automatically filtered based on security. If a user does not have access to a particular resource, it is not displayed in the navigation when it is rendered on the page.

7. In the Custom Content Attributes section, specify any desired display options for the resource. For more information, see Section 11.3.9, "How to Set Display Options for a Resource."

   Note: Typically for custom content, you will not need to specify any attributes, because the attributes for the resources generated by the content provider will be specified by the content provider.

8. In the Custom Content Parameters section, define the parameters on the folder. The parameters you define here are passed to and interpreted by the `CustomContentProviderV2` implementation.

9. Save the navigation model definition file.

11.3.8 How to Organize Resources in a Navigation Model

You can add folders to your navigation model to create subsets of resources or to group similar resources. You can also add separators to divide the navigation model into distinct groupings of resources. Folders and separators enable you to break up the items in the navigation model when it is rendered on in a page, making it easier to locate items.

You can also rearrange items within the navigation model by dragging them up and down in relation to each other and dragging them inside other items to indicate hierarchy.

To organize resources:

1. In the Application Navigator, right-click the navigation model whose resources you want to organize, and choose **Open**.
2. To add a folder or separator:
   a. In the Design view for the navigation model, in the Navigation column on the left side, select the node in the navigation model under which you want to add the folder or separator.
   b. Click the **Add new node** icon and choose **Folder** or **Separator**.
   c. The **Id** field is automatically populated with a generated ID. You can keep this ID, but you will probably want to change it to something slightly more descriptive. The ID must be unique within the navigation model.

      If the resource is at the top level of the navigation model, the ID must not be `wc_navtitle` or `wcnav_externalId`; these IDs are reserved.

      Tip: If the ID is not unique, or you enter any invalid characters in the ID, the border of the field becomes red.

   d. In the **Visible** field, specify whether the resource is displayed in the navigation. The default value is the EL expression `#{true}`, which means the resource is displayed to all users at all times. Enter `false` or `#{false}` to hide the resource from all users at all times, or enter an EL expression to specify the conditions under which the resource is displayed. If the field is empty, the value defaults to `true`.
3. To move a resource within the navigation model, select the item and drag it to the desired position.
4. To indent one resource (child item) under another (parent item), select the child item and drag it over the top of the parent item.
5. To re-parent a resource, right click it and choose **Change Parent**.
6. Save the navigation model definition file.

11.3.9 How to Set Display Options for a Resource

You can specify various display options for resources to determine their appearance and behavior when the navigation is rendered on a page. The display options available depend on the type of the resource.

To set display options for a resource:

1. In the Application Navigator, right-click the navigation model that contains the resource for which you want to set display options, and choose **Open**.

2. In the Design view for the navigation model, in the Navigation column on the left side, select the resource.
3. In the URL Attributes panel, click the **Add** icon and choose the attribute that you want to set. Table 11.2 lists the available attributes.

Table 11–2 Resource Attributes

Attribute	Description
Title	The title displayed for the resource when the navigation model is rendered on a page.
AccessKey	A key mnemonic (single character) that users can enter to access the resource without using the mouse.
Description	A description of the resource.
IconURI	An icon to visually represent the resource. This is displayed next to the Title when the navigation model is rendered on a page.
Subject	Keywords to facilitate searching of the resource.
Target	The location on the container page where the resource is displayed when it is selected, either in the same browser window (`_self`), a new window (`_blank`), or a popup (`_popup`), or any other location supported by the navigation UI. **Note:** Popups are not supported for pages.
ToolTip	Text that displays to provide additional information about the resource when users hover the mouse over the Title.
Modified	The date of the last modification of the resource. This attribute is used for site map creation.
ChangeFrequency	How frequently the resource is likely to change: `always`, `hourly`, `daily`, `weekly`, `monthly`, `yearly`, `never`. This attribute is used for site map creation.
Significance	The priority of this resource relative to other resources in the navigation model, within the range `0.0` to `1.0`. This attribute is used for site map creation.
ExternalId	An ID to enable a direct reference to a node in the navigation model from a static link in the page. Nodes in the default navigation model with this attribute defined can be accessed using `/faces/wcnav_externalId/`*`externalId`* from an `af:goLink` component in a page or using `/wcnav_externalId/`*`externalId`* from an External Link navigation item in the navigation model. The alternative is to access the node by its hierarchy, which is faster, but may change as the application develops. To directly access a node in a non-default navigation model, you must specify the path of the navigation model by setting the wcnav.modelPath URL parameter, for example: `/faces/wcnav_externalId/myNavigationItem?wcnav.modelPath=/oracle/webcenter/siteresources/scopedMD/s7f446cab_f622_4b68_a83e_b7eaf28b52ec/navigation/gsr0271c712_721a_4565_9f0e_755784a7093b/myProjectNavigationModel.xml`

4. From the **Value Type** dropdown list, choose:
   - **Literal String**—to specify a string as the value for the attribute

- **Resource Bundle**—to use a resource bundle to provide localized values for the attribute.

5. In the **Display Value** field, enter the value for the attribute. If you are using a resource bundle, click the **Browse** icon to select the resource bundle to search for the localized value.
6. Save the navigation model definition file.

11.4 Selecting the Default Navigation Model

The default navigation model provides a convenient way to select a navigation model that should be used by default by an application. Page template designers can then reference this default navigation model without having to know its actual name. For example, in navigation EL expressions, the default navigation model is often referenced, such as in `#{navigationContext.defaultNavigationModel}`.

When you first create a Framework application, the seeded navigation model, `default-navigation-model.xml`, is set as the default navigation model. If you subsequently create your own navigation model, rather than having to explicitly reference this navigation model whenever you want to use it, you can set it as the default.

You can set the default navigation model for an application by editing the `oracle.webcenter.portalapp.navigation.model` preference in the `adf-config.xml` file.

To select the default navigation model:

1. In the Application Resources pane of the Application Navigator, right-click the **adf-config.xml** file, and choose **Open**.

   Tip: To locate the `adf-config.xml` file, expand the **Descriptors** node, and then the **ADF META-INF** node.

2. Click the **Source** tab.
3. Locate the ADF preference with the following `id`:

   ```
   oracle.webcenter.portalapp.navigation.model
   ```

4. Set the `value` attribute to the path of the navigation model that you want to use as the default for the portal, for example:

   ```
   value="oracle/webcenter/portalapp/navigations/myNavigationModel.xml"
   ```

   Example 11–1 shows an example of the complete preference element.

Example 11–1 The Default Navigation Model ADF Preference

```
<portal:adf-portal-config>
  <preferences>
    ...
    <preference id="oracle.webcenter.portalapp.navigation.model"
        desc="Default Navigation Model"
        value="oracle/webcenter/portalapp/navigations/myNavigationModel.xml"
        resourceType="navigation" display="true"/>
    ...
  </preferences>
</portal:adf-portal-config>
```

5. Save the `adf-config.xml` file.

11.5 Editing the Navigation Renderer

The navigation renderer is a JSPX page renders non-page navigation resources, including:

- Portlets
- Content
- External links
- External applications
- Task flows

To change how these resources are rendered, you can edit the navigation renderer page. For example, you may also want to add an image before or after the navigation renderer task flow or change from a flow to a stretch layout.

The navigation renderer page is hidden by default, so you must first show it in your application before you can edit it.

Note: In the vast majority of cases, you should be able to edit the existing navigation renderer to meet your requirements. However, if required, you can create your own navigation renderer page. For example, the navigation renderer expects a `content` facet. If you want to use a different facet, you can create your own page and place the navigation renderer task flow in the appropriate location for your template.

If you choose to create your own navigation renderer page, it must include a navigation renderer region that contains the navigation renderer task flow.

To set your page as the navigation renderer for the application, edit the navigation renderer ADF preference (`oracle.webcenter.portalapp.navigation.renderer`) to point to the appropriate file. For detailed instructions for how to edit an ADF preference, see Section 11.4, "Selecting the Default Navigation Model."

To edit the navigation renderer:

1. In the Application Navigator, right-click the application project and choose **Project Properties**.
2. Expand the **Project Source Paths** node and select **Web Application**.
3. Select the **Excluded** tab.
4. Select the navigation renderer file (`navigation-renderer.jspx`) and click **Remove** to remove it from the list of excluded files.
5. Click **OK**.

   The navigation renderer JSPX file is now listed in the Application Navigator under the following node:

   `/oracle/webcenter/portalapp/pages/`

6. Right click **navigation-renderer.jspx** and choose **Open**.
7. Click the **Source** tab and edit the file as required.
8. When you are finished, save the `navigation-renderer.jspx` file.

11.6 Setting the REDIRECT_OPTIONS Context Parameter

If your application does not use partial page refresh (PPR), the URL in the browser changes whenever a user navigates to a resource in the navigation model. However, the URL shown by the browser does not reflect the current page; it shows the URL of the last page shown.

To enable the browser to display the URL of the current page, you can set the `oracle.webcenter.navigationframework.REDIRECT_OPTIONS` context parameter in the `web.xml` file.

> **Note:** Setting the `oracle.webcenter.navigationframework.REDIRECT_OPTIONS` context parameter affects performance by adding an extra request/response for every navigation.

To set the REDIRECT_OPTIONS context parameter:

1. In the Application Navigator, right-click the **web.xml** file and choose **Open**.
2. In the Overview Editor, expand the Context Initialization Parameters section.
3. Click the Create Context Initialization Parameter icon.
4. In the **Name** field, enter `oracle.webcenter.navigationframework.REDIRECT_OPTIONS`.
5. In the **Value** field, enter `toPrettyURL`.

   This adds the following to the `web.xml` file:

```
<context-param>
 <param-name>oracle.webcenter.navigationframework.REDIRECT_OPTIONS</param-name>
 <param-value>toPrettyURL</param-value>
</context-param>
```

6. Save the `web.xml` file.

11.7 Enabling Runtime Administration of Navigation Models

The Resource Manager in a Framework application enables users with administrative privileges to create and modify portal resources, including navigation models, at runtime. By exposing navigations in the Resource Manager, you allow administrators to manage these navigations at runtime. If you want to create and expose more navigations after deploying the application, you can export them from JDeveloper to the runtime application.

> **Note:** If you create a navigation model at runtime and do not import it back into the design time environment, your new navigation model may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To expose a navigation in the Resource Manager, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

You can also control exactly which resources users can add to navigation models at runtime by editing the navigation registry file, `navigation-registry.xml`.

For more information about the runtime features of navigation models, see the chapter "Working with Navigations" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This section includes the following subsections:

- Section 11.7.1, "How to Edit the Navigation Registry File"
- Section 11.7.2, "How to Enable Further Development of Deployed Navigation Models"

11.7.1 How to Edit the Navigation Registry File

The navigation registry is a Resource Catalog specifically for registering which resources users can add to navigation models at runtime. Each Framework application contains one navigation registry file, `navigation-registry.xml`, which is created when you create the application. You can edit this file to change which resources to include. For example, if you do not want users to be able to embed other navigation models at runtime, you can remove the Navigation Reference resource from the navigation registry.

To edit the navigation registry:

1. In the Application Navigator, right-click the `navigation-registry.xml` file.

   Tip: You can find this file under *`Application_Root`*`/oracle/webcenter/portalapp/navigations`.

2. Edit the file the same way as any other Resource Catalog. For more information, see Section 16.2.3, "How to Manage Content in the Resource Catalog."

   You can add the following resources to the navigation registry:

   - Folder
   - Link
     - Task Flow
     - Portlet
     - Content
     - Other
   - Resource Catalog
   - Component
   - Custom Folder
   - Custom Content

   You can also modify and remove existing resources in the navigation registry, and rearrange resources in the registry.

3. When you are finished, save the `navigation-registry.xml` file.

11.7.2 How to Enable Further Development of Deployed Navigation Models

Seeded navigation models cannot be modified or deleted in the Resource Manager. Users with administrative privileges can make a copy of a seeded navigation model and edit or delete it, or they can create their own navigation models.

If users create or edit navigation models at runtime, you may find that you need to bring them back into JDeveloper for further enhancement. You can do this by downloading the navigation model from the deployed application, editing it in JDeveloper, and uploading it back to the application. At design time, this type of round-trip development includes importing the downloaded navigation model, editing it in JDeveloper, and exporting it so that it can be uploaded back to the deployed application.

In addition, if you create a new navigation model after you have deployed your application, you can export the navigation model from JDeveloper and upload it into the deployed application.

This section describes how to import and export navigation models in JDeveloper. It includes the following subsections:

- Section 11.7.2.1, "Importing a Navigation Model"
- Section 11.7.2.2, "Exporting a Navigation Model"

For more information about round-trip development, see Section 17.4, "Enabling Round-Trip Development of Resources."

11.7.2.1 Importing a Navigation Model

When you download a navigation model from the Resource Manager, an EAR file is created. On importing the EAR file in JDeveloper, the files from the EAR file are extracted to the folder `*Application_Root*/Portal/public_html/oracle/webcenter/siteresources/scopedMD/*scope_GUID*`. You can open the navigation model and edit it as you would any other navigation model at design time. The updated navigation model must then be exported to the deployed application.

For detailed steps to import a resource, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

11.7.2.2 Exporting a Navigation Model

You would need to perform an export for the following reasons:

- To expose a new navigation model in an already deployed application.
- To copy back a navigation model that you imported from a deployed application and modified.

When you export a navigation model from JDeveloper, an EAR file is created. Users with administrator privileges can then upload this EAR file to a deployed application using the Resource Manager.

For detailed steps to export a resource, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

11.8 Working with WebCenter Portal: Spaces Navigation Models

Out of the box, WebCenter Portal: Spaces provides seeded navigation models that you can use in your spaces. If these default navigation models do not suit your requirements, you can build your own navigation models within WebCenter Portal:

Spaces. For more information, see the section "Working with Navigation" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

However, WebCenter Portal: Spaces may not provide all the controls to create a navigation model that provides all the functionality that you require. In such cases, in JDeveloper you can edit navigation models originally created in WebCenter Portal: Spaces, and then upload them back into WebCenter Portal: Spaces.

WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, such as navigation models, to WebCenter Portal: Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build custom navigation models and other types of resources for WebCenter Portal: Spaces.

When working with WebCenter Portal: Spaces navigation models in JDeveloper, you should consider the following:

- All the resources required for a WebCenter Portal: Spaces navigation model exist within WebCenter Portal: Spaces. Therefore, you should really only use JDeveloper to restructure WebCenter Portal: Spaces navigation models and perform minor edits. First, you should create as much of the navigation model in WebCenter Portal: Spaces as possible and add the required resources. You can then use JDeveloper to refine the order of the resources and set all required attributes and parameters.
- If you do want to use JDeveloper to add resources to a WebCenter Portal: Spaces navigation model, you should obtain the resource ID from WebCenter Portal: Spaces by accessing the About dialog for the resource. For more information, see the section "Viewing Information About a Resource" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- When adding resources to a WebCenter Portal: Spaces navigation model, you must manually enter the path to the resource. You cannot use the Browse icon. For example, to enter a link to a page, you must identify the ID of the page in WebCenter Portal: Spaces and then enter that value manually as `page://`*`pathToXmlFile`*.

12

Designing the Look and Feel of Your Portal

When building a portal, it is important to decide on a look and feel for your portal. You may also want to ensure that the look and feel is consistent across the portal. You can achieve this kind of consistency across portal pages by using page templates and page styles.

This chapter provides information about creating and managing page templates and page styles. It also explains how to create mashup styles that can be used to create task flows at runtime. It includes the following subsections:

- Section 12.1, "Introduction to Page Templates and Styles"
- Section 12.2, "Working with Page Templates"
- Section 12.3, "Working with Page Styles"
- Section 12.4, "Working with Mashup Styles"

12.1 Introduction to Page Templates and Styles

Page templates define how individual pages and groups of pages display on a user's screen. When you use page templates to display pages, the pages are always consistent in structure and layout across the application. For example, a Side Navigation - Flow page template provides a navigation panel on the left side of the page, and the size of every component in the main content area is calculated based on the size of its children.

Page styles, on the other hand, determine the layout, and at times the initial content, for a new page in your portal. For example, the Web Page style provides a one-column layout and enables users to expose external web content in their application; a Blank page style provides a one-column layout and has few restrictions on the types of content users can add to the pages that are based on it.

Page templates and page styles are JSPX pages that you can create and publish in JDeveloper. While page templates can be used to create portal pages at design time and runtime, page styles are used only when creating pages at runtime. This section describes the different page templates and styles that you can create at design time. It includes the following subsections:

- Section 12.1.1, "Page Templates"
- Section 12.1.2, "Page Styles"
- Section 12.1.3, "Mashup Styles"
- Section 12.1.4, "Content Presenter Display Templates"

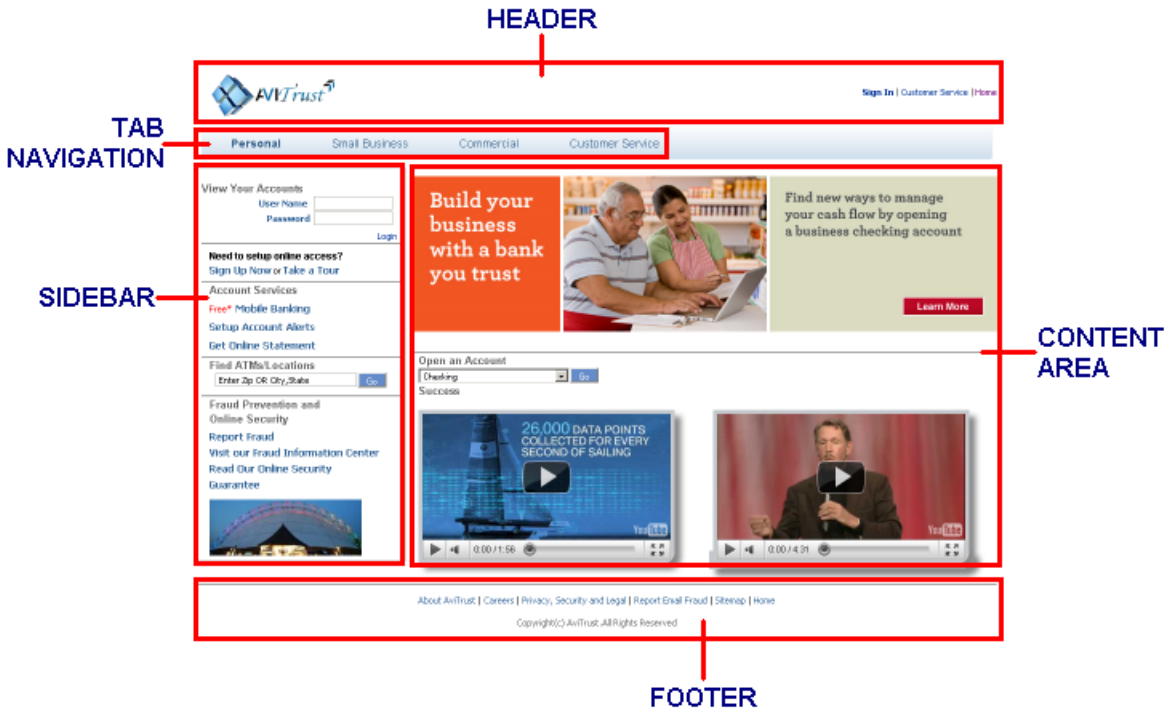
Note: Skins also contribute to the look and feel of your portal by specifying the application background color, screen fonts, and, with some skins, the shapes and images used for application buttons and icons. However, skins are not discussed in this chapter. For information, see Chapter 15, "Creating and Managing Skins."

12.1.1 Page Templates

A page template typically defines the layout and default navigation model for pages in a portal application. It provides a uniform look and feel. Figure 12–1 shows a sample portal based on a page template, which provides the following:

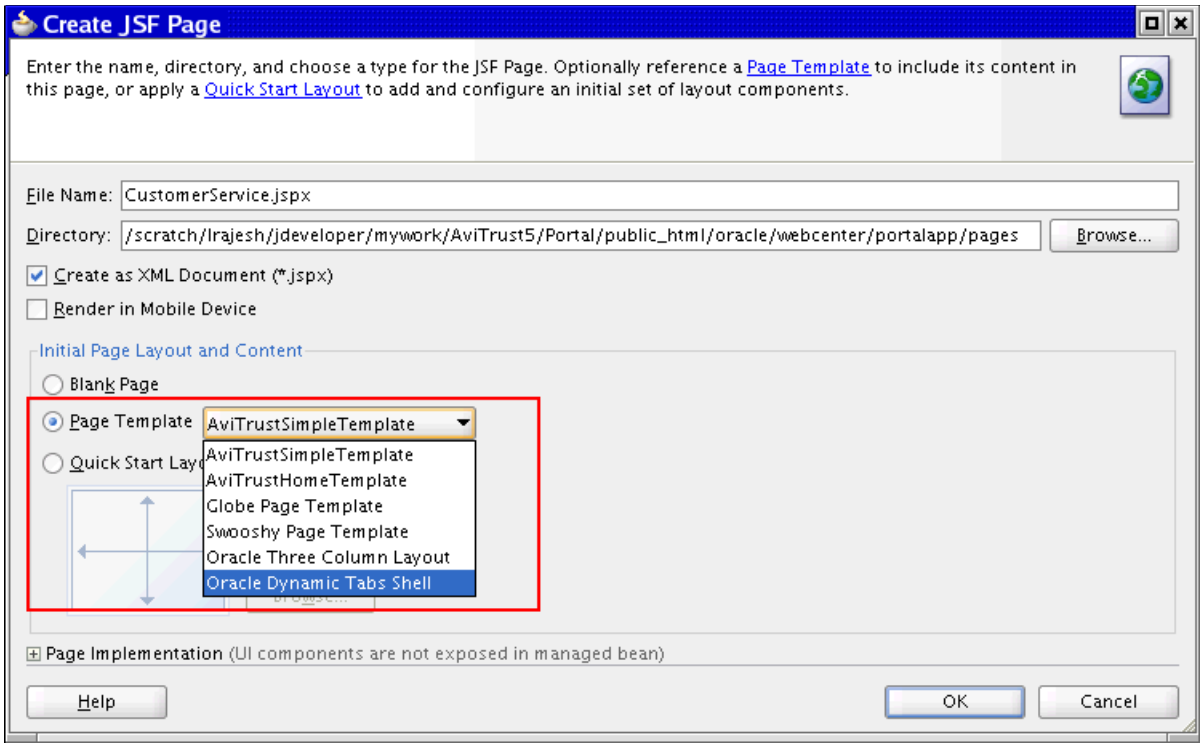
- Header with the company logo and the Sign In, Customer Service, and Home links.
- Default tab navigation to access different resources in the portal.
- Sidebar with a login task flow and some useful links.
- Footer with a copyright statement and a few disclaimer statements.
- A central area for adding content.

Figure 12–1 Sample Portal that Uses a Page Template



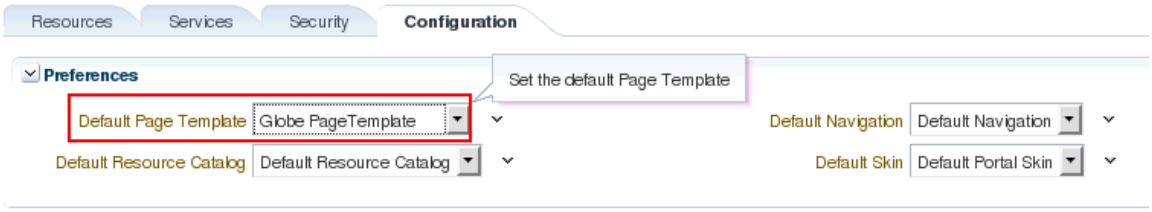
Page templates are reference-based and use inheritance. That is, if you change the page template, then all pages that reference that template automatically inherit the change. You can create and publish page templates in JDeveloper, and they can be used to create pages both at design time and runtime. At design time, you can choose a page template in the Create JSF Page dialog, shown in Figure 12–2.

Figure 12–2 Page Template Option in the Create JSF Page Dialog



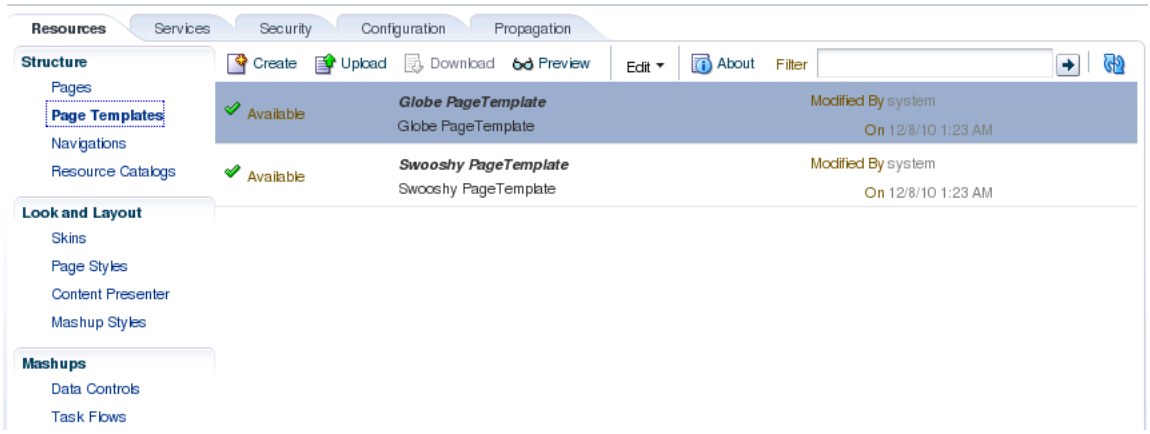
At runtime, you can apply a page template by selecting it in the application's Configuration page, as shown in Figure 12–3.

Figure 12–3 Page Template Option in the Application's Configuration Page



An administrator or delegate can manage the exposed page templates in the Resource Manager, shown in Figure 12–4.

Figure 12–4 Page Templates in the Resource Manager



For more information, see Section 12.2, "Working with Page Templates."

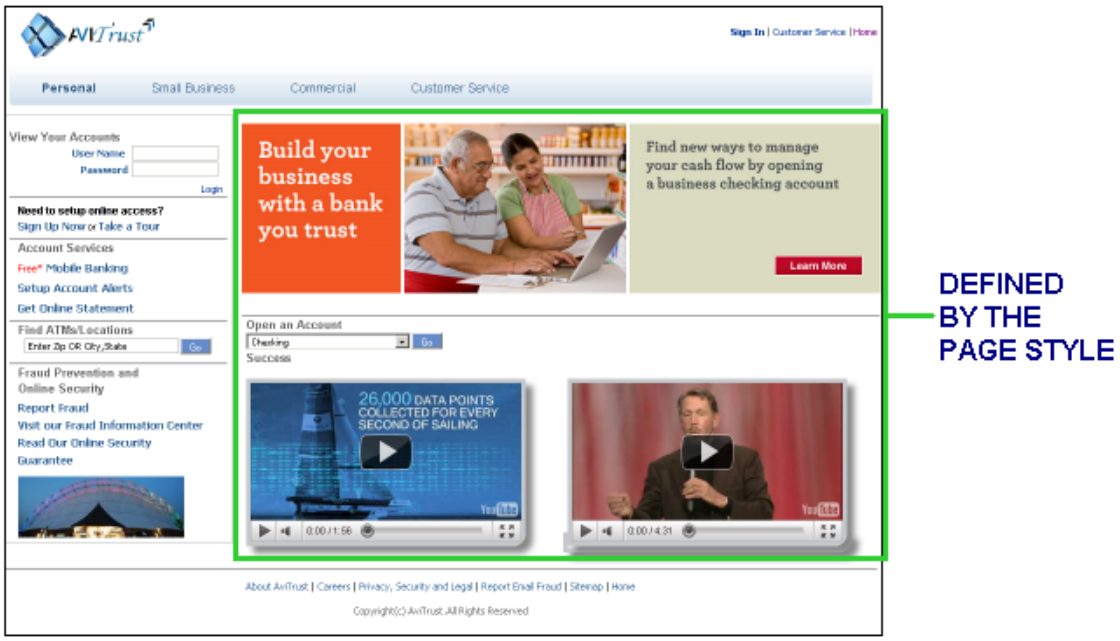
12.1.2 Page Styles

A page style is a JSPX page used for pages created at runtime. It describes the layout of a newly created page and may also dictate the type of content that the page supports.

When a user creates a page using a page style, the layout and initial content are copied from the page style to the newly created page. Unlike page templates, page styles are not reference-based, that is, if you change a page style, the change is not inherited in pages that use the style.

Typically, a page style contains components that enhance the usefulness and appearance of a given page. These include an in-place HTML text editor, images, layout boxes, hyperlinks, and so on. Content contributors can further populate the page with content. Figure 12–5 shows a sample portal page that is based on a page style.

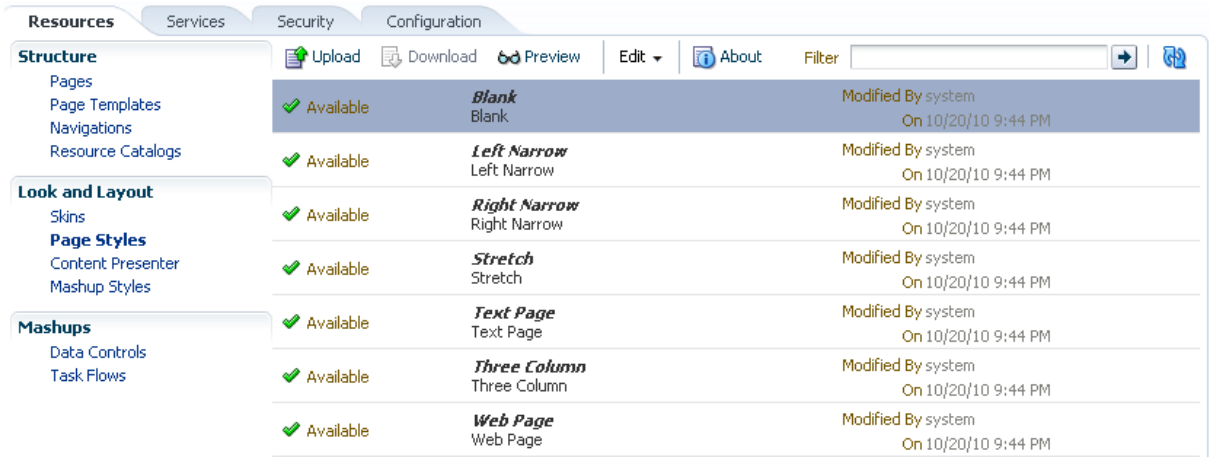
Figure 12–5 Sample Page that Uses a Page Style



You create page styles in JDeveloper, but they are only used at runtime to create pages. Page styles that you expose in the application can be used and managed as follows:

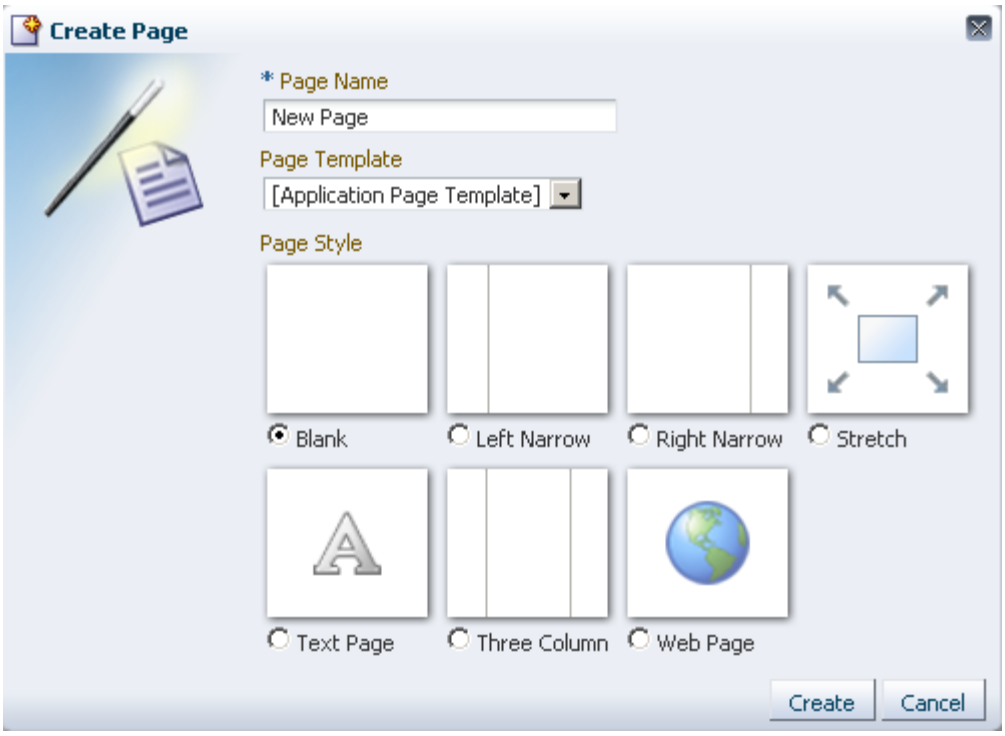
- An administrator or delegate can manage the exposed page styles in the Resource Manager, shown in Figure 12–6.

Figure 12–6 Page Styles in the Resource Manager



- A user can create pages based on the available page styles. The Create Page option is available in the Resource Manager. It is also available in other application pages that contain the `Page - Create New` task flow. When a user clicks the Create Page option, the Create Page dialog displays a set of predefined styles, as shown in Figure 12–7. The user can choose a style and create a page based on that. As the layout is already in place in the new page, the user only needs to add content to different areas of the page.

Figure 12–7 Create Page Dialog with Default Styles



For more information about creating page styles in JDeveloper, see Section 12.3, "Working with Page Styles." For information about managing page styles at runtime, see "Working with Page Styles" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

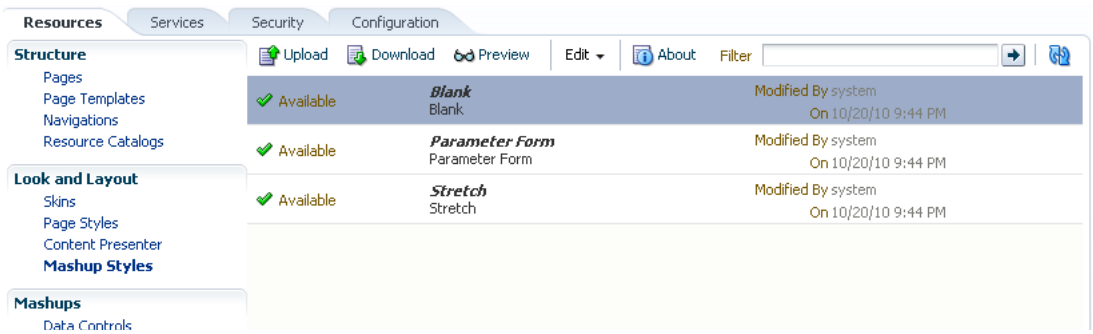
12.1.3 Mashup Styles

Mashup styles are not very different from page styles, except that they are used at runtime to create task flows, also known as business mashups.

Mashup styles are used only at runtime to create mashups, but they can be created and published in JDeveloper. For more information, see Section 12.4, "Working with Mashup Styles."

Only an administrator or delegate can create and manage the exposed mashup styles in the runtime Resource Manager, shown in Figure 12–8.

Figure 12–8 Mashup Styles in the Resource Manager



Mashup styles are displayed in the Create New Task Flow dialog box (Figure 12–9), which is invoked by clicking Create on the Resource Manager - Task Flows page. The user can choose a style and create a mashup based on that style. For more information, see "Creating and Managing Task Flows" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Figure 12–9 Create New Task Flow Dialog with Default Styles



12.1.4 Content Presenter Display Templates

Content Presenter display templates are used to define how Content Presenter renders content items on a Framework application page. Similar to page styles and mashup styles, Content Presenter templates can also be managed at runtime in the Resource Manager. These templates are not discussed in this chapter. For details, see Chapter 29, "Creating Content Presenter Display Templates."

12.2 Working with Page Templates

You can create any number of custom page templates and expose them to users in the Create JSF Page dialog. This section provides information about creating page templates and enabling runtime administration of page templates. It includes the following subsections:

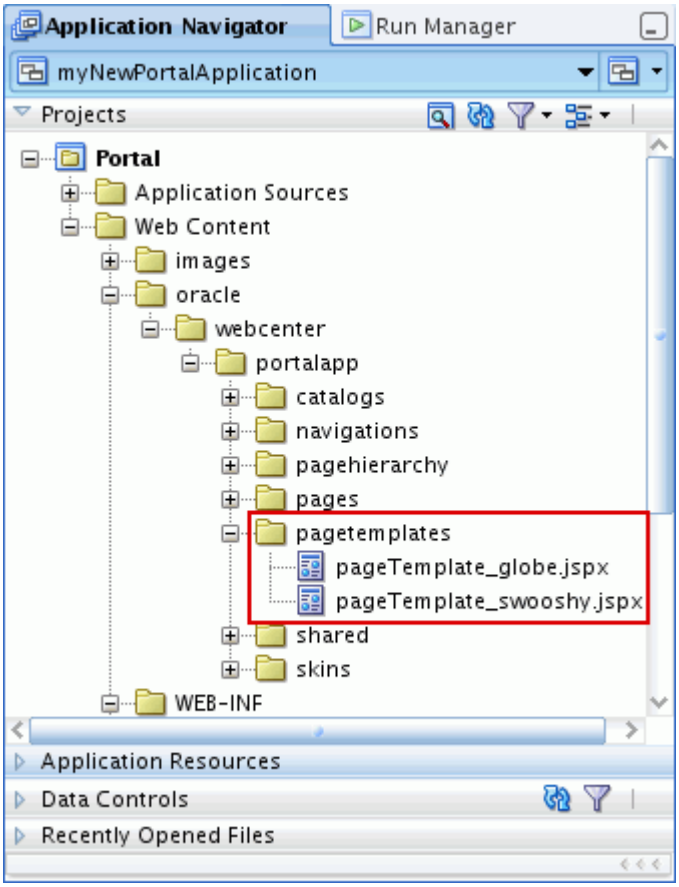
- Section 12.2.1, "Introduction to the WebCenter Portal Application Seeded Page Templates"
- Section 12.2.2, "How to Create a Page Template"
- Section 12.2.3, "What You Should Know About Creating Page Templates for WebCenter Portal Applications"
- Section 12.2.4, "Enabling Runtime Switching of Page Templates"
- Section 12.2.5, "How to Edit a Page Template"
- Section 12.2.6, "How to Delete a Page Template"
- Section 12.2.7, "How to Enable Runtime Administration of a Page Template"
- Section 12.2.8, "How to Enable Further Development of Deployed Page Templates"

- Section 12.2.9, "How to Create and Manage WebCenter Portal: Spaces Page Templates"

12.2.1 Introduction to the WebCenter Portal Application Seeded Page Templates

When you create an application using the WebCenter Portal Application template, and select the **Configure the application with standard Portal features** option, two seeded templates are added to your application: `pageTemplate_globe.jspx` and `pageTemplate_swooshy.jspx`.

Figure 12–10 The Seeded Page Templates in the Application Navigator

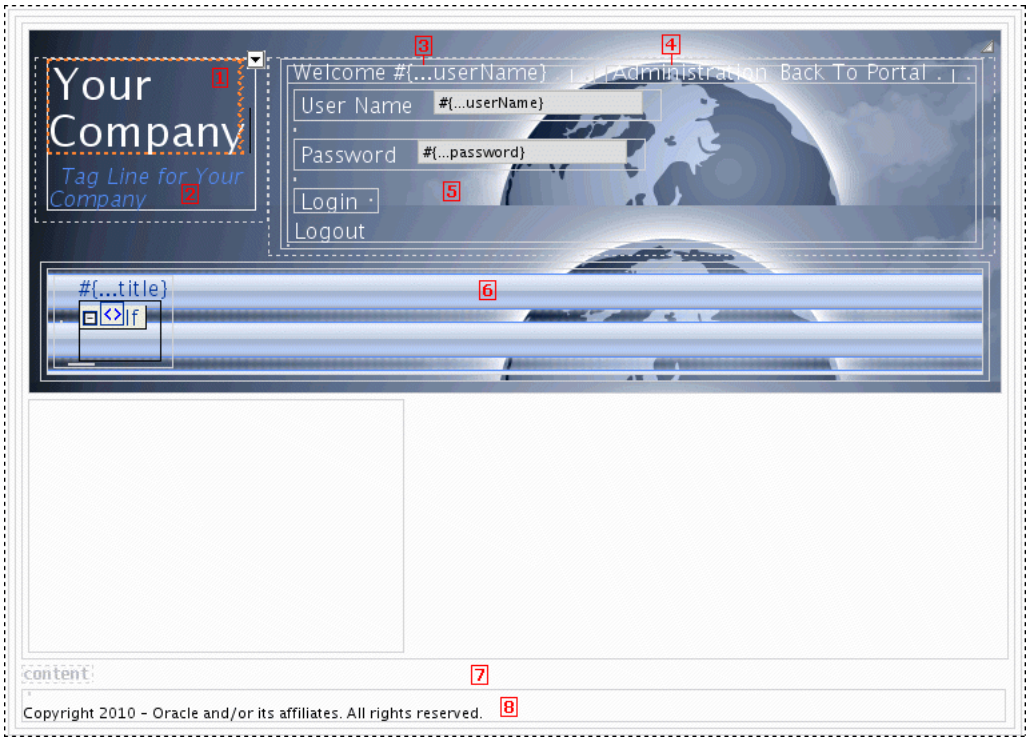


These templates both offer essentially the same functionality but with different graphics. They include the following:

1. Link to the portal home page
2. Tag line
3. Welcome message
4. Link to the seeded Administration page
5. Login area that converts to a logout link when users are logged in
6. Navigation bar
7. Area for adding content to pages based on the template
8. Copyright notice

Figure 12–11 shows the `pageTemplate_globe.jspx` page template with each of the above features called out.

Figure 12–11 The pageTemplate_globe.jspx Seeded Page Template



You can edit this default page template to meet your particular requirements, or, more typically, you can create your own. If you choose to create your own page template, the seeded page templates are still a useful tool for you to discover the different things you can achieve in your own templates.

12.2.2 How to Create a Page Template

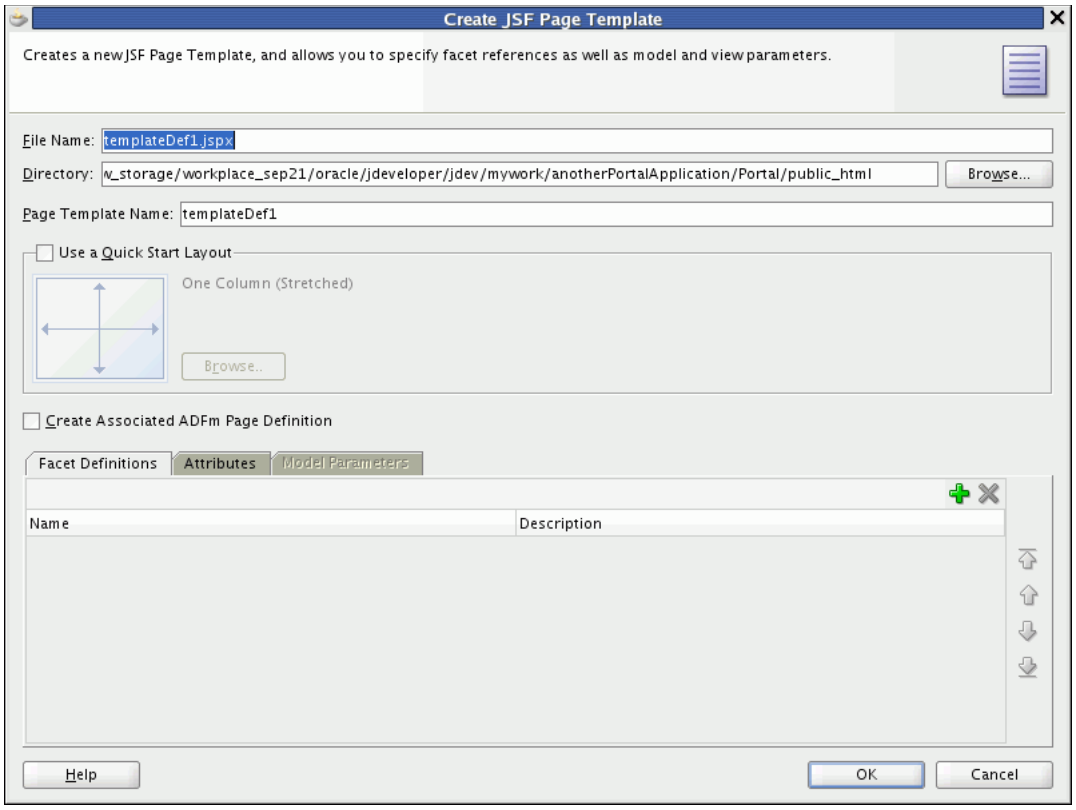
The following section focuses on the aspects of page templates that are specific to WebCenter Portal applications. For general information about JSF page templates, see the section "Using Page Templates" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

To create a page template:

1. In the Application Navigator, right-click the node where you want to create the page template (for example, `/oracle/webcenter/portalapp/pagetemplates`) and choose **New**.
2. In the New Gallery, expand **Web Tier**, select **JSF** and then **JSF Page Template**, and click **OK**.
3. In the Create JSF Page Template dialog (Figure 12–12), in the **File Name** field, enter a name for the JSPX file that represents the page template, for example `myCompanyPageTemplate.jspx`.

   The file name identifies the page template in the Application Navigator.

Figure 12–12 The Create JSF Page Template Dialog



4. In the **Directory** field, enter the full directory path of the location under which to create the page template.

   Note: In a WebCenter Portal application, to expose the new page template in the runtime Resource Manager, you must create the page template in the *Application_Root*`/Portal/public_html/oracle/webcenter/portalapp` directory.

   ---

5. In the **Page Template Name** field, enter a display name for the page template.

   The display name is shown in the **Page Template** dropdown list of the Create JSF Page wizard to enable users to select this template to use for a newly created page. For this reason, you should make the page template name something that helps users quickly identify which type of pages this template should be used for.

6. Select **Use a Quick Start Layout** if you want to use one of the predefined layouts for your page template.

7. Select **Create Associated ADFm Page Definition** to use ADFm Model data bindings in the page template.

   Associating a page definition with the page template enables you to include objects that have model elements to them (for example, task flows and portlets) in the page template. It also enables users to switch to a different page template at runtime.

Note: The seeded page templates all have associated page definitions, therefore it is highly recommended that you select this option when creating page templates in WebCenter Portal applications. Within an application page templates must either all have associated page definitions or none have associated page definitions.

If you have a combination of page templates with and without associated page definitions, users will not be able to switch templates at runtime.

After selecting this option, you can click the **Model Parameters** tab and click the **New** icon to define the model parameters for data bound components set with the ADFm declarative data binding layer. The values for these parameters can be set at runtime by the calling page.

Note: By default, the page template definition name is based on the page template name. Do not create the page definition with a different name.

8. Click the **Facet Definitions** tab to define the areas in the page template where content can be inserted to pages built using the template. A page template that is intended for use in a WebCenter Portal application must contain at least one facet definition called `content`.
   a. Click the **New** icon to add a new row to the **Facet Definitions** tab.
   b. In the **Name** field for the new facet definition, enter `content`.
   c. In the **Description** field, enter a brief description for the facet definition, for example `Main content area`.
   d. Create additional facet definitions as required.
   e. For information about how to add the facets to the required location in the page template, see step 12.

   Example 12–1 shows the `content` facet definition in the `<af:xmlContent>` section of the `pageTemplate_globe.jspx` seeded WebCenter Portal application page template.

Example 12–1 Content Facet in the Globe Seeded Page Template

```
<af:xmlContent>
  <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
    <display-name>Globe Page Template</display-name>
    <facet>
      <description>Facet for content</description>
      <facet-name>content</facet-name>
    <facet>
    ...
  </component>
</af:xmlContent>
```

9. Click the **Attributes** tab and click the **New** icon to define attributes for passing values from pages that use the template to the template itself. These values can be

used by the page template to drive what is actually displayed on individual pages that use the template.

For example, the `pageTemplate_globe.jspx` seeded WebCenter Portal application page template includes a link to the seeded Administration page. If the current page *is* the Administration page, then this link does not need to be displayed. The template uses an attribute, called `isAdminPage`, to identify whether the current page is the seeded Administration page. The value of this attribute, set by pages that use the template, determines whether or not to display the link to the Administration page.

Example 12–2 shows the `isAdminPage` attribute definition in the page template.

Example 12–2 Attribute Definition in the Globe Seeded Page Template

```
<af:xmlContent>
  ...
    <attribute>
      <attribute-name>isAdminPage</attribute-name>
      <attribute-class>java.lang.Boolean</attribute-class>
      <default-value>#{false}</default-value>
    </attribute>
  ...
</af:xmlContent>
```

Example 12–3 shows how the page template uses the value of the `isAdminPage` attribute to determine whether or not to render the link to the seeded Administration page.

Example 12–3 Conditional Value for Rendering the Administration Page Link

```
<af:pageTemplateDef var="attrs">
  ...
  <af:goLink id="pt_glnk1" text="Administration"
             destination="/admin"
             rendered="#{attrs.showAdmin !attrs.isAdminPage}"
             inlineStyle="font-size:small; color:White;"/>
```

Example 12–4 shows how the seeded Administration page sets the `isAdminPage` attribute to `true`, ensuring that the page template does not render the link when users are viewing the Administration page.

Example 12–4 Setting the Value of a Page Template Attribute

```
<af:pageTemplate value="#{bindings.pageTemplateBinding.templateModel}" id="pt1">
  <f:attribute name="isAdminPage" value="#{true}"/>
```

The seeded page templates include the following additional attributes:

```
<attribute>
  <attribute-name>contentWidth</attribute-name>
  <attribute-class>java.lang.String</attribute-class>
  <default-value>960px</default-value>
</attribute>

<attribute>
  <attribute-name>showNavigation</attribute-name>
  <attribute-class>java.lang.Boolean</attribute-class>
  <default-value>#{true}</default-value>
</attribute>
```

```
<attribute>
  <attribute-name>showGreetings</attribute-name>
  <attribute-class>java.lang.Boolean</attribute-class>
  <default-value>#{securityContext.authenticated}</default-value>
</attribute>

<attribute>
  <attribute-name>showLogin</attribute-name>
  <attribute-class>java.lang.Boolean</attribute-class>
  <default-value>#{true}</default-value>
</attribute>

<attribute>
  <attribute-name>showAdmin</attribute-name>
  <attribute-class>java.lang.Boolean</attribute-class>
  <default-value>
    #{WCSecurityContext.userInAppRole['Administrator']}
  </default-value>
</attribute>
```

10. Click **OK**.

    The page template is created and opened in the visual editor.

11. Drag components from the Component Palette and drop them onto the page template in the visual editor.

    Navigation UI is a very important aspect of a page template. For information about adding navigation UI to your page template, see Chapter 14, "Visualizing Your Portal Navigation."

    For information about adding other resources to your page template, for example, portlets, task flows, or content, see Chapter 13, "Adding Resources to Your Portal."

    Other features you might want to provide in your page template include a login/logout area or a search field. The seeded page templates include an example of a login form.

    Throughout your page templates, you can use expression language (EL) expressions to specify a variable value instead of a constant value. For descriptions of common EL expressions, see Appendix G, "Expression Language Expressions."

12. For areas on the page template where users can place their own content in pages based on the template:

    a. Drag the **FacetRef** component from the Component Palette and drop it onto the appropriate location on the page template.

    b. In the Insert Facet Ref dialog, from the **Facet Name** dropdown list, select the facet that you want to use for this area of the page template. If you enter a new facet name, JDeveloper creates a new facet definition in the page template definition file.

    Example 12–5 shows how the content facet is included in the `pageTemplate_globe.jspx` seeded WebCenter Portal application page template.

Example 12–5 The content Facet in the Globe Seeded Page Template

```
<af:pageTemplateDef var="attrs">
  <af:panelGroupLayout id="pt_root" layout="scroll">
    <af:panelGroupLayout id="pt_pg12" layout="vertical">
      ...
      <af:facetRef facetName="content"/>
```

```
      ...
    </af:panelGroupLayout>
  </af:panelGroupLayout>
  ...
<af:pageTemplateDef>
```

12.2.3 What You Should Know About Creating Page Templates for WebCenter Portal Applications

When creating page templates for use in WebCenter Portal applications, consider the following:

- To ensure that the page template can be exposed in the Resource Manager, create the page template JSPX file inside the `/oracle/webcenter/portalapp` directory. Out of the box there is already a `pagetemplates` subdirectory in which you can store your page templates.
- The seeded page templates are a valuable source of examples of features that you can include in your own page templates. Even if you do not intend to base your page templates on the seeded ones at all, it is still worth studying them for ideas. For example, the seeded page templates include a login form that you might want to reuse. The seeded templates also include a good example of navigation UI.
- In the Create Page Template dialog, you must select the **Create Associated ADFm Page Definition** option. If you do not select this option, users will not be able to switch between page templates at runtime. For page template switching to work, one of the following conditions must be met:
  - All page templates in the application have associated page definitions.
  - None of the page templates in the application have associated page definitions.

  Therefore, because the seeded page templates provided with the WebCenter Portal Application template have associated page definitions, for page template switching to work, any other page templates created within the application must also have associated page definitions for page template.
- Your page template must include at least one facet definition, and that facet definition must be called `content`.
- If the navigation model referenced by the page template includes External Link navigation items that use `mailto:` links, you must explicitly handle these in the page template. Example 12–6 uses JSTL to inspect the navigation item's `externalURL` to see if it starts with the string `mailto:`. If it does, an ADF Faces `goLink` component is used to render the link.

Example 12–6 Rendering a mailto: Link

```
...
<c:choose>
  <c:when test="${fn:startsWith(node.externalURL, 'mailto:')}">
    <af:goLink id="pt_gl_mail" text="#{node.title}"
               destination="#{node.externalURL}"/>
  </c:when>
  <c:otherwise>
    ...
  </c:otherwise>
</c:choose>
...
```

- If your page template includes a Content Presenter task flow to display a content item, you must manually modify the task flow properties to display the content, as follows:
  - `datasource`: *`connection_name`*#dDocName:*`content_ID`*
  - `datasourceType`: `dsTypeSingleNode`
  - `taskFlowInstId`: A unique identifier

  For more information about Content Presenter task flow properties, see Table 31–4, " Content Presenter Task Flow Parameters".
- If you are making a page template available for runtime management, any dependent objects of the page template, such as images, JavaScripts, style sheets, or HTML files, must be stored under the following directory, which is the default content directory for page templates created at design time:

  ```
  Application_Root/Portal/public_html/oracle/webcenter/portalapp/shared
  ```

  In addition, as a best practice, it is recommended to store dependent objects in resource-specific subdirectories under the `shared` directory. For example, images for a page template named `Template1` could be stored in:

  ```
  /oracle/webcenter/portalapp/shared/pageTemplates/Template1
  ```

  This avoids any potential conflict when additional resources are imported from runtime.

  When adding the resource as a portal resource, you must ensure that the **Content Directory** points to the appropriate `shared` directory.

  To reference dependent objects in your page template, use the following format:

  ```
  /oracle/webcenter/portalapp/shared/objectFileName
  ```

  For example:

  ```
  <af:image
  source="/oracle/webcenter/portalapp/shared/pageTemplates/Template1/logo.gif"
  id="pt_12">
  ```

12.2.4 Enabling Runtime Switching of Page Templates

When you create a page, you select a specific page template to use for that page. This adds a static reference to the page template to the JSPX file for the page. For example:

```
<af:pageTemplate
    viewId="/oracle/webcenter/portalapp/pagetemplates/pageTemplate_globe.jspx"
    value="#{bindings.pageTemplateBinding}"
    id="pt1">
  <f:facet name="content"/>
</af:pageTemplate>
```

A binding of the page template is created in the page's page definition file. For example:

```
<executables>
  <variableIterator id="variables"/>
  <page path="oracle.webcenter.portalapp.pagetemplates.pageTemplate_globePageDef"
        id="pageTemplateBinding" Refresh="ifNeeded"/>
</executables>
```

To use a different page template for the page, you must change both these references.

You can also change these references to use EL expressions to enable runtime switching of the page template.

> **Note:** You can switch between page templates at runtime only if the page template was created with an associated page definition.

To enable runtime switching of page templates:

1. In the Application Navigator, right-click the JSPX file for the page, and choose **Open**.
2. Click the **Source** tab.
3. Replace the following code:

```
<af:pageTemplate viewId="pathToPageTemplate"
                 value="#{bindings.pageTemplateBinding}"
                 id="ptId">
  <f:facet name="content"/>
</af:pageTemplate>
```

   with:

```
<af:pageTemplate value="#{bindings.pageTemplateBinding.templateModel}"
                 id="ptId">
  <f:facet name="content"/>
</af:pageTemplate>
```

4. Right-click the JSPX file and choose **Go to Page Definition**.
5. Click the **Source** tab.
6. Replace the following code:

```
<page path="pathToPageTemplatePageDef"
      id="pageTemplateBinding" Refresh="ifNeeded"/>
```

   with:

```
<page viewId="${preferenceBean.defaultPageTemplate}"
      id="pageTemplateBinding" Refresh="ifNeeded"/>
```

7. In the Application Resources pane of the Application Navigator, right-click the `adf-config.xml` file, and choose **Open**.

   > **Tip:** To locate the `adf-config.xml` file, expand the **Descriptors** node, and then the **ADF META-INF** node.

8. Click the **Source** tab.
9. Locate the ADF preference with the following id:

```
oracle.webcenter.portalapp.pagetemplate.pageTemplate
```

10. Set the `value` attribute to the page template to apply to the page. For example:

```
<portal:preference
  id="oracle.webcenter.portalapp.pagetemplate.pageTemplate"
  desc="Default Page Template"
  value="/oracle/webcenter/portalapp/pagetemplates/pageTemplate_globe.jspx"
```

```
resourceType="Template" display="true"/>
```

12.2.5 How to Edit a Page Template

You can make changes to your page template after its initial creation. Any changes you make are automatically rolled out to any pages that use the page template.

To edit a page template:

1. Locate the page template that you want to edit.
2. Right-click the page template and choose **Open**.
3. In the visual editor make any changes to the layout and content of the page template.
4. To add new facet definitions or attributes, select the `af:pageTemplateDef` tag in the Structure window and use the Property Inspector.
5. Save the page template definition file.

   If iterative development is enabled, any changes that you make to page templates can be seen immediately in the running application. For more information about iterative development, see Section 1.5, "Preparing for Iterative Development."

12.2.6 How to Delete a Page Template

If a page template is no longer required within your portal, you can delete it. When you delete a page template, attempting to run any page that has a direct reference to the page template will produce exceptions.

To delete a page template:

1. Locate the page template that you want to delete.
2. Right-click the page template and choose **Delete**.
3. The Confirm Delete dialog identifies if the page template is used within the application. Click **Show Usages** for more details.

   You can use this information to determine whether you really want to delete the template.
4. Click **Yes** to continue and delete the page template. Click **No** to cancel the operation.

12.2.7 How to Enable Runtime Administration of a Page Template

The Resource Manager in a WebCenter Portal application enables users with administrative privileges to create and modify portal resources, including page templates, at runtime. By exposing page templates in the Resource Manager, you allow administrators to manage these templates at runtime. If you want to create and expose more page templates after deploying the application, you can export them from JDeveloper to the runtime application.

Note: If you create a page template at runtime and do not import it back into the design time environment, your new page template may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To expose a page template in the Resource Manager, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

For more information about the runtime features of page templates, see the chapter "Working with Page Templates" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

12.2.8 How to Enable Further Development of Deployed Page Templates

Seeded page templates cannot be modified or deleted in the Resource Manager. However, users with administrative privileges can make a copy of a seeded page template and edit or delete it, or they can create their own page templates.

If users create or edit page templates at runtime, you may find that you need to bring them back into JDeveloper for further enhancement. You can do this by downloading the page template from the deployed application, editing it in JDeveloper, and uploading it back to the application. At design time, this type of round-trip development includes importing the downloaded page template, editing it in JDeveloper, and exporting it so that it can be uploaded back to the deployed application.

In addition, if you create a new page template after you have deployed your application, you can export the page template from JDeveloper and upload it into the deployed application.

This section describes how to import and export page templates in JDeveloper. It includes the following subsections:

- Section 12.2.8.1, "Importing a Page Template"
- Section 12.2.8.2, "Exporting a Page Template"

For more information about round-trip development, see Section 17.4, "Enabling Round-Trip Development of Resources."

12.2.8.1 Importing a Page Template

When you download a page template from the Resource Manager, an EAR file is created. On importing the EAR file in JDeveloper, the files from the EAR file are extracted to the folder `*Application_Root*/Portal/public_html/oracle/webcenter/siteresources/scopedMD/scope_GUID`. You can open the page template and edit it as you would edit any other page template at design time. The updated page template must then be exported to the deployed application.

For detailed steps to import a resource, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

12.2.8.2 Exporting a Page Template

You would need to perform an export for the following reasons:

- To expose a new page template in an already deployed application.
- To copy back a page template that you imported from a deployed application and modified.

When you export a page template from JDeveloper, an EAR file is created. Users with administrator privileges can then upload this EAR file to a deployed application using the Resource Manager. Once uploaded, the page template can be managed and used to create application pages.

For detailed steps to export a resource, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

12.2.9 How to Create and Manage WebCenter Portal: Spaces Page Templates

Out of the box, WebCenter Portal: Spaces provides seeded page templates that you can use in your spaces. If these default page templates do not suit your requirements, you can build your own page templates within WebCenter Portal: Spaces. For more information, see the chapter "Working with Page Templates" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

However, WebCenter Portal: Spaces may not provide all the controls to create a page template that provides all the functionality that you require. In such cases, you can create your own page template in JDeveloper and then upload it to WebCenter Portal: Spaces. You can also download an existing page template from WebCenter Portal: Spaces, edit it in JDeveloper, or use one of the sample page templates as your starting point, and then upload it back into WebCenter Portal: Spaces.

WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, such as page templates, to WebCenter Portal: Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build custom page templates and other types of resources for WebCenter Portal: Spaces. For details, see Section 56, "Extending the Spaces Application Using JDeveloper."

When creating or editing page templates in JDeveloper for use in WebCenter Portal: Spaces, consider the following:

- The page template must include a facet definition called `content`.
- The page template must include at least one UI component with `class="WCPGTEMPLATE"`. The runtime editing of skins enables the targeting of the template background in a basic tab; it depends on this style being there.

12.3 Working with Page Styles

Page styles are JSPX pages. You can create any number of custom page styles and expose these to users in the Create Page dialog. This section provides information about creating page styles and enabling runtime administration of page styles. It includes the following sections:

- Section 12.3.1, "How to Create a Page Style"
- Section 12.3.2, "How to Enable Runtime Administration of a Page Style"
- Section 12.3.3, "How to Enable Further Development of Deployed Page Styles"
- Section 12.3.4, "How to Create and Manage Page Styles for Spaces"

12.3.1 How to Create a Page Style

You can either create a page style from scratch or use an ADF page template to create it, then design it with the required layout components. However, there are a few prerequisites for creating a page style to be used in a custom application.

Requirements for Page Styles Exposed as Portal Resources

- To ensure that the style can be exposed in the Resource Manager, create the page style JSPX file inside the `/oracle/webcenter/portalapp` directory. You can

either create a subdirectory for page styles, for example `pagestyles`, or save the page style in the `pages` directory that is available out-of-the-box.

A page style can be exposed in the runtime Resource Manager only if it is present within the `/oracle/webcenter/portalapp/` structure.

- If a page style is based on a page template, then while designing the page style JSPX, add the `content` facet, or any other facet defined in the page template, to contain page content.
- To make the page customizable, add a `Page Customizable` component within the content facet. As the `Page Customizable` contains a child `Panel Customizable` component by default, users can add content to the page at runtime.
- To enable dynamic selection of the ADF template to be used for the page style, in the page style JSPX, specify an EL value for `pageTemplate`. Set the `value` attribute with an EL expression to a managed bean method that returns the page template ID, as shown in the following example:

  ```
  <af:pageTemplate value="#{bindings.pageTemplateBinding.templateModel}" id="T">
  ```

  This is useful in scenarios where the ADF template to be used is determined based on criteria such as the current user or scope.
- You may want to ensure that a new page created from a page style has the same look and feel as the portal it resides in. For this, you can configure the page style to use an ADF template based on the currently-applied page template. To achieve this, you must insert an EL reference to the page template's page definition in the page style's page definition file, as shown in the following example:

  ```
  <page viewId="#{preferenceBean.defaultPageTemplateViewId}"
        id="pageTemplateBinding" Refresh="ifNeeded"/>
  ```

- Create a managed bean with a method that returns the currently-applied page template ID, as shown in the following example:

  ```
  public class siteTemplatesManager {
      final private String templateA = "/templateA.jspx";
      final private String templateB = "/templateB.jspx";
      private String currentSiteTemplateViewId;

      public siteTemplatesManager() {
          super();
          currentSiteTemplateViewId = templateA;
      }
      public String gettemplateViewId() {
          return currentSiteTemplateViewId;
      }
      public void settemplateAViewId(ActionEvent ae) {
          currentSiteTemplateViewId = templateA;
      }
      public void settemplateBViewId(ActionEvent ae) {
          currentSiteTemplateViewId = templateB;
      }
  }
  ```

Section 19.6.1, "How to Create Templates for Pages Created at Runtime" describes in detail how to create a page style.

Example 12–7 shows the code for a sample page style, `/oracle/webcenter/portalapp/pages/pageStyles/TemplateFlowingTwoC`

olumn.jspx, which creates a basic page designed to flow and provide two columns that are 35 percent and 65 percent in proportion. Example 12–8 shows the associated page definition file, /oracle/webcenter/portalapp/pages/pageStyles/TemplateFlowingTwoColumnPageDef.jspx.

> **See Also:** Section 19.3.4.2, "Custom Styles" for information about where custom page styles and their page definitions can be stored and how to ensure that these styles show up in the Create Page dialog at runtime.

Example 12–7 Source Code of a Custom Page Style

```
<?xml version='1.0' encoding='utf-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
xmlns:f="http://java.sun.com/jsf/core"
xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
xmlns:trh="http://myfaces.apache.org/trinidad/html" version="2.1">
  <jsp:directive.page deferredSyntaxAllowedAsLiteral="true"/>
  <jsp:directive.page contentType="text/html;charset=utf-8"/>
  <f:view>
    <af:document title="#{pageDocBean.title}" id="docrt">
      <af:form usesUpload="true" id="f1">
        <af:pageTemplate value="#{bindings.pageTemplateBinding.templateModel}"
        id="T">
          <f:facet name="content">
            <pe:pageCustomizable id="pc11">
              <af:panelGroupLayout id="pgl1" layout="scroll"
              styleClass="replace_with_scheme_name"
              inlineStyle="replace_with_inline_style">
                <trh:tableLayout id="tl1" width="100%">
                  <trh:rowLayout id="rl1">
                    <trh:cellFormat id="cf1" width="35%" valign="top">
                      <cust:panelCustomizable id="hm_pnc1" layout="scroll"/>
                    </trh:cellFormat>
                    <trh:cellFormat id="cf2" width="65%" valign="top">
                      <cust:panelCustomizable id="hm_pnc2" layout="scroll"/>
                    </trh:cellFormat>
                  </trh:rowLayout>
                </trh:tableLayout>
                <trh:tableLayout id="tl2"/>
              </af:panelGroupLayout>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:pageTemplate>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

Example 12–8 Page Definition of the Custom Page Style

```
<pageDefinition version="11.1.1.41.30" id="TemplateFlowingTwoColumnPageDef"
  Package="oracle.webcenter.siteresources.scopedMD.s8bba98ff_4cbb_40b8_beee_
296c916a23ed.pageStyle.gsr2b052c53_0eba_43fd_81e2_7f12dbb885a5">
```

```
  <parameters/>
  <executables>
    <page viewId="#{preferenceBean.defaultPageTemplate}"
          id="pageTemplateBinding" Refresh="ifNeeded"/>
    <taskFlow id="pageeditorpanel"
taskFlowId="#{pageEditorBean.pageEditorPanel}"/>
  </executables>
</pageDefinition>
```

12.3.2 How to Enable Runtime Administration of a Page Style

By exposing page styles in the Resource Manager, you allow administrators to manage them at runtime. You can create and export additional page styles even after deploying the application, without having to redeploy it. At runtime, the Resource Manager provides an option to upload the page styles that you exported.

> **Note:** If you create a page style at runtime and do not import it back into the design time environment, your new page style may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To expose a page style in the Resource Manager, it must have a valid entry in the application's `generic-site-resources.xml` file. Use the following options on the context menu of a page style JSPX file to register or deregister a page style in the Resource Manager:

> **Note:** The Resource Manager-specific options are displayed on the context menu only if you created the page style in the `/oracle/webcenter/portalapp` directory.

- **Create Portal Resource**: Creates an entry for the page style in the `generic-site-resources.xml` file.

  While creating an entry, you must set the attributes described in Table 12–1. The attributes in this table are specific to page styles. For the generic steps that are common to all resources, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

Table 12–1 Page Style Properties

Property	Description
Display Name	The display name for the page style.
Resource Type	The type of resource being exposed. This is a read-only field that is populated automatically based on certain internal validations on the JSPX file.

Table 12–1 (Cont.) Page Style Properties

Property	Description
Icon URL	The path to the image that you want to display alongside the page style name in the Create Page dialog. This is an optional setting. Specify the absolute path to the image file. For example, for a page style that is imported from WebCenter Portal: Spaces, the Icon URL is of the following format: *APPLICATION_ROOT*/oracle/webcenter/siteresources/scopedMD/shared/*FILE_NAME*
Description	A description for the page style.
Content Directory	The directory containing dependent objects of the page style, such as images, JavaScripts, style sheets, or HTML files. Objects associated with a resource are rendered only if they are stored in the `/oracle/webcenter/portalapp/shared` directory.

- **Update Portal Resource**: Edit the attributes that you defined while creating page style entries. For more information, see Section 17.2.3, "How to Update Portal Resource Properties."
- **Remove Portal Resource**: Remove the entry for the selected page style from the `generic-site-resources.xml` file. For more information, see Section 17.2.4, "How to Remove a Resource from the Resource Manager."

12.3.3 How to Enable Further Development of Deployed Page Styles

Seeded page styles cannot be modified or deleted at runtime. However, users with administrator privileges can make a copy of a seeded page style and edit its properties or delete it.

The Edit Source option in the Resource Manager enables users to modify the content of a copied or custom page style at runtime. However, as text editing is easier in JDeveloper than in the runtime source editor, users may choose to download a page style, edit it in JDeveloper, and upload it back to the deployed application. At design time, this type of round-trip development includes importing the downloaded page style, editing it in JDeveloper, and exporting it so that it can be uploaded back to the deployed application.

This section describes how to import and export page styles in JDeveloper.

12.3.3.1 Importing a Page Style

When you download a page style from the Resource Manager, an EAR file is created containing the following objects:

- Page style JSPX file
- Page definition file
- The `generic-site-resources.xml` file
- The content directory referenced by the page style. By default, this directory is `oracle/webcenter/siteresources/scopedMD/shared`. The EAR file will not contain this folder if the shared directory has no content.

The `generic-site-resources.xml` file contains metadata that helps identify the type of resource being imported. The metadata is also used when you export the page style back to the runtime application.

On importing the EAR file in JDeveloper, the files from the EAR are extracted to the folder *`Application_Root`*`/Portal/public_html/oracle/webcenter/siteresources/scopedMD/<`*`scope_GUID`*`>`. You can open the page style and edit it as you would edit any other page at design time. The updated page style must then be exported to the deployed application.

> **Note:** If you imported a page style from WebCenter Portal: Spaces but want to use it in a Framework application, ensure that you delete the WebCenter Portal: Spaces-specific privilege mapping from the page definition file. This means that you remove the `<permission>` element from the style's page definition. You may get errors if you retain the privilege mapping.

For the detailed steps to import a resource, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

12.3.3.2 Exporting a Page Style

You would need to perform an export for the following reasons:

- To expose a new page style in an already deployed application.
- To copy back a page style that you imported from a deployed application and modified.

When you export a page style from JDeveloper, an EAR file is created containing the page style JSPX, its page definition file, the `generic-site-resources.xml` file, and a `shared` folder containing the images associated with the page style. Users with administrator privileges can then upload this EAR file to a deployed application using the Resource Manager. Once uploaded, the page style can be managed and used to create application pages.

For the detailed steps to import a resource, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

12.3.4 How to Create and Manage Page Styles for Spaces

Out of the box, Spaces provides several seeded page styles. If these default styles do not suit your requirements, you can build your own page styles within Spaces. For more information, see the chapter "Working with Page Styles" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

However, Spaces may not provide all the controls to create a page style that provides all the functionality that you require. In such cases, you can create your own page style in JDeveloper and then upload it to Spaces. You can also download an existing page style from Spaces, edit it in JDeveloper, or use one of the sample page styles as your starting point, and then upload it back into Spaces. In JDeveloper, you can create a page style just like any other JSPX page and upload it to Spaces.

Oracle WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, such as page styles, to Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the

`WebCenterSpacesResources` project to build custom page styles and other types of resources for Spaces.

The primary tasks of creating and editing page styles are the same for Framework applications and Spaces applications, but you must make a few additional considerations for Spaces applications. This section covers the special considerations for page styles to be consumed in Spaces.

Requirements for Page Styles Used in Spaces

Use the `CustomPagePermission` class to map permissions explicitly for the page style, as shown in the following example:

```
<permission
permissionClass="oracle.webcenter.page.model.security.CustomPagePermission"
target="ps_targetusage">
  <privilege-map operation="administer" privilege="manage"/>
  <privilege-map operation="create" privilege="create"/>
  <privilege-map operation="delete" privilege="delete"/>
  <privilege-map operation="edit" privilege="update"/>
  <privilege-map operation="personalize" privilege="personalize"/>
  <privilege-map operation="view" privilege="view"/>
</permission>
```

For the general page style prerequisites, see "Requirements for Page Styles Exposed as Portal Resources". For the procedure to create a page style, see Section 12.3.1, "How to Create a Page Style."

12.4 Working with Mashup Styles

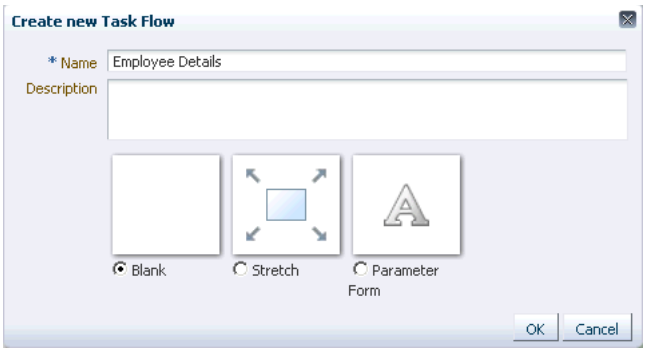
Mashup styles are ADF task flows that can be used as templates to create new task flows at runtime. However, a mashup style can have only a single view fragment. In addition, if the mashup style refers to code, for example, using an EL value, then that code must be present in the deployed application.

You may choose to create mashup styles for the following purposes:

- To provide a specific, consistent layout for task flows.
- To seed a Parameter Form or UI picker that can in turn be used in the task flows created from the mashup style, for example, a mashup style with a space picker drop down. When a task flow based on this style is populated with an Activity Stream task flow, the picker drop down can be wired to an input parameter on the Activity Stream task flow. The Activity Stream task flow then displays activities specific to the selected space.
- To provide a predefined UI that can be bound separately to data, for example, you can create a flowing data layout in the mashup style, create a task flow from it, drop a data control, then bind the flowing data layout to the data control.

The Resource Manager provides a Task Flows page in which users with administrative privileges can create and manage task flows. When a user clicks Create on the Task Flows page, The Create new Task Flow dialog, shown in Figure 12–13, displays the seeded mashup styles available to users.

Figure 12–13 Seeded Mashup Styles



Note: Managing ADF Task Flows

Just like how you manage mashup styles at runtime, you can create and manage task flows also in the Resource Manager. The Resource Manager - Task Flows page enables users to create, edit, delete, import, and export task flows.

You cannot export a custom task flow created in JDeveloper as a portal resource, but you can import a task flow created at runtime, modify it, and export it back to the runtime application. The steps to import, modify, and export task flows are the same as those for mashup styles. Therefore, the export, import, and edit tasks in this section are applicable for task flows and mashup styles.

For information about adding and exporting custom task flows, see Section 13.3, "Adding Custom Task Flows to Your Portal."

This section describes how to create mashup styles and configure them to be exposed at runtime. It contains the following tasks:

- Section 12.4.1, "How to Create a Mashup Style"
- Section 12.4.2, "How to Enable Runtime Administration of Mashup Styles"
- Section 12.4.3, "How to Enable Further Development of Mashup Styles in a Deployed Application"
- Section 12.4.4, "How to Create and Manage Mashup Styles for Spaces"

12.4.1 How to Create a Mashup Style

Creating a mashup style is the same as creating a task flow, but you create mashup styles only so that they can be used as templates to create task flows at runtime. Mashup styles can be consumed at runtime only if you expose them in the Resource Manager.

Requirements for Mashup Styles Exposed as Portal Resources

To expose a mashup style in the Resource Manager, you must consider the following requirements when creating a mashup style in JDeveloper:

- Create the task flow definition XML file inside the `/oracle/webcenter/portalapp` directory. You can either create a subdirectory for mashup styles, for example `mashupstyles`, or save the mashup style directly in the `portalapp` directory.

- Create *only one* view fragment for a task flow that will be used as a mashup style, and create a page definition for this JSFF view fragment.
- Add a `Panel Customizable` component to the view fragment from the Composer tag library. This ensures that task flows based on this mashup style can be populated at runtime.

Example 12–9 shows the code for a sample mashup style that enables records from a bug details database to be listed in the Activity Stream format. The style is created based on knowledge of the data source being accessed and the columns available in the data source. In this example:

- `dataPresenter.dummyData.collectionModel` is a null data source that must be replaced with the appropriate data control reference when the mashup style is used at runtime.
- `#{row.`*`COLUMN`*`}` entries are placeholders that will reference data columns from the data controls when the mashup style used.
- `#{empty row.COLUMN ? '#{row.COLUMN}' : row.COLUMN}` is used to display `#{row.COLUMN}` in the task flow created using this mashup style until `dataPresenter.dummyData.collectionModel` is replaced and data is properly bound. Using a construct like this ensures that task flows created from this mashup style give users a hint about the available data slots.

Example 12–9 Sample Mashup Style

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <af:panelGroupLayout id="pgl1">
    <cust:panelCustomizable id="pc1">
      <table border="0" width="100%">
        <af:iterator id="i1"
                     value="#{dataPresenter.dummyData.collectionModel}"
                     var="row">
        <tr class="PortletText1">
          <td width="16px">
            <af:image id="img1"

source="http://abc.mycompany.com:80/webcenter/content/conn/UCM/Public/images/wiki/
#{row.IMAGE}"/>
          </td>
          <td>
           <af:outputText id="otCol1"
                          value="#{empty row.PLAYER1 ? '#{row.PLAYER1}' :
row.PLAYER1}"
                          inlineStyle="color:#333333; font-size:12px;
font-weight:bold;"/>
           <af:spacer id="spacer1" width="5px"/>
           <af:outputText id="otCol2"
                          value="#{empty row.MESSAGE ? '#{row.MESSAGE}' :
row.MESSAGE}"
                          inlineStyle="color:#333333; font-size:12px;"/>
          <af:spacer id="spacer2" width="5px"/>
          <af:goLink id="gl1"
                       destination="https://bug.mycompany.com/pls/bug/webbug_
edit.edit_info_top?rptno=#{empty row.RPTNO ? '#{row.RPTNO}' : row.RPTNO}"
```

```
                        targetFrame="_blank"
                        text="#{empty row.RPTNO ? '#{row.RPTNO}' : row.RPTNO"/>
          </td>
        </tr>
        <tr>
          <td></td>
          <td>
            <af:outputText id="otCol3"
                           value=""#{empty row.BUG_SUBJECT ? '#{row.BUG_SUBJECT}'
: row.BUG_SUBJECT}"
                           inlineStyle="color:#666666; font-size:12px;"/>
          </td>
        </tr>
        <tr>
          <td></td>
          <td>
            <af:outputText id="otCol4"
                           value="#{empty row.TIMEOFDAY ? '#{row.TIMEOFDAY}' :
row.TIMEOFDAY}"
                           inlineStyle="color:#666666; font-size:10px;"/>
          </td>
        </tr>
        </af:iterator>
      </table>
    </cust:panelCustomizable>
  </af:panelGroupLayout>
</jsp:root>
```

For detailed information about ADF task flows, see "Creating ADF Task Flows" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

12.4.2 How to Enable Runtime Administration of Mashup Styles

By exposing mashup styles in the Resource Manager, you allow administrators to manage these styles and create task flows based on these styles. You can also export selected mashup styles from JDeveloper so that administrators can modify or delete them at runtime.

Managing mashup styles at runtime is similar to managing page styles. For more information, see "Working with Page Style Resources" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

> **Note:** If you create a mashup style at runtime and do not import it back into the design time environment, your new mashup style may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To expose a mashup style in the Resource Manager, it must have a valid entry in the application's `generic-site-resources.xml` file. Use the following options on the context menu of a task flow definition XML file to register or deregister the mashup style in the Resource Manager:

Note: The Resource Manager-specific options are displayed on the context menu only if you created the mashup style in the `/oracle/webcenter/portalapp` directory.

- **Create Portal Resource**: Creates an entry for the mashup style in the `generic-site-resources.xml` file.

  While creating an entry, you must set the attributes described in Table 12–2. The attributes in this table are specific to mashup styles. For the generic steps that are common to all resources, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

Table 12–2 Mashup Style Properties

Property	Description
Display Name	The display name for the style.
Resource Type	The type of resource being exposed. You must select `Mashup Style` here.
Icon URL	The path to the image that you want to display alongside the mashup style name in the Create a new Task Flow dialog. This is an optional setting. Ensure that the image is in the class path and can be accessed at runtime.
Description	A description for the mashup style.

- **Update Portal Resource**: Edit the attributes that you defined while creating an entry for the mashup style. For more information, see Section 17.2.3, "How to Update Portal Resource Properties."
- **Remove Portal Resource**: Remove the entry for this mashup style from the `generic-site-resources.xml` file. For more information, see Section 17.2.4, "How to Remove a Resource from the Resource Manager."

12.4.3 How to Enable Further Development of Mashup Styles in a Deployed Application

Seeded mashup styles cannot be modified or deleted in the Resource Manager. However, users with administrator privileges can make a copy of a seeded mashup style and edit its properties or delete it.

The Edit Source option in the Resource Manager enables users to modify the content of a copied or custom mashup style at runtime. However, as text editing is easier in JDeveloper than in the runtime source editor, users may choose to download a mashup style, edit it in JDeveloper, and upload it back to the deployed application. At design time, this type of round-trip development includes importing the downloaded mashup style, editing it in JDeveloper, and exporting it so that it can be uploaded back to the deployed application.

This section describes how to export and import mashup styles in JDeveloper.

12.4.3.1 Importing Mashup Styles

When you download a mashup style from the Resource Manager, an EAR file is created containing the following objects:

- Task flow definition file
- View fragment JSFF file

- View fragment page definition file
- The `generic-site-resources.xml` file
- The content directory referenced by the mashup style. By default, this directory is `oracle/webcenter/siteresources/scopedMD/shared`. The EAR file will not contain this folder if the `shared` directory has no content

The `generic-site-resources.xml` file contains metadata that helps identify the type of resource being imported. This metadata is also used when you export the mashup style back to the runtime application.

On importing the EAR file in JDeveloper, the files from the EAR are extracted to the folder *`Application_Root`*`/Portal/public_html/oracle/webcenter/siteresources/scopedMD/<`*`scope_GUID`*`>`. You can open the mashup style and edit it as you would edit any other task flow at design time. The updated mashup style must then be exported to the runtime application.

Note: When updating a runtime mashup style in JDeveloper, if the icon URL points to a location in the application classpath, you may see an error message prompting you to place the icon in the *`APPLICATION_ROOT`*`/oracle/webcenter/siteresources/scopedMD` directory.

To proceed with updating the mashup style, you can provide a null value for the icon URL, and replace the correct value at runtime after uploading the mashup style.

For the detailed steps to import a resource, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

12.4.3.2 Exporting Mashup Styles

You would need to perform an export for the following reasons:

- To expose a new mashup style in an already deployed application.
- To copy back a mashup style that you imported from a deployed application and modified.

When you export a mashup style from JDeveloper, an EAR file is created containing the task flow definition file, the view fragment JSFF, its page definition file, the `generic-site-resources.xml` file, and a `shared` folder containing the image used for the mashup style. Users with administrator privileges can then upload this EAR file to a deployed application using its Resource Manager. Once uploaded, the mashup style can be managed and used to create task flows.

For the detailed steps, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

12.4.4 How to Create and Manage Mashup Styles for Spaces

Out of the box, Spaces provides several seeded mashup styles. If these default styles do not suit your requirements, you can build your own mashup styles within Spaces. For more information, see the chapter "Managing Mashup Styles" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

However, Spaces may not provide all the controls to create a mashup style that provides all the functionality that you require. In such cases, you can create your own

mashup style in JDeveloper and then upload it to Spaces. You can also download an existing mashup style from Spaces, edit it in JDeveloper, or use one of the sample mashup styles as your starting point, and then upload it back into Spaces.

Oracle WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, such as mashup styles, to Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build custom mashup styles and other types of resources for Spaces.

The primary tasks of creating and editing mashup styles are the same for Framework applications and Spaces applications. For the procedure to create a mashup style, see Section 12.4.1, "How to Create a Mashup Style."

For the general mashup style prerequisites, see "Requirements for Mashup Styles Exposed as Portal Resources".

13

Adding Resources to Your Portal

When building up the content of your WebCenter Portal: Framework application, you can bring in content from various resources, including:

- Pages
- Portlets and pagelets
- WebCenter Portal service task flows and custom task flows
- Content from UCM
- External applications

Adding content to your Framework application is no different from adding content to any other web application.

This chapter includes the following sections:

- Section 13.1, "Adding Pages to Your Portal"
- Section 13.2, "Adding Service Task Flows to Your Portal"
- Section 13.3, "Adding Custom Task Flows to Your Portal"
- Section 13.4, "Adding Content from Content Server to Your Portal"
- Section 13.5, "Adding Portlets to Your Portal"
- Section 13.6, "Adding Pagelets to Your Portal"
- Section 13.7, "Adding External Applications to Your Portal"
- Section 13.8, "Adding Data Controls to Your Portal"

13.1 Adding Pages to Your Portal

A portal is made up of pages that provide the structure of the portal and expose the content. If you create your portal using WebCenter Portal's Framework application template, a home page and several other pages are created for you. You can create your own pages to construct the rest of your portal.

For more information, see Section 5.3, "Adding Pages to a Portal."

After creating the pages, you can organize them into a page hierarchy to determine the structure of the pages. The page hierarchy also determines the security for those pages.

For more information, see Section 69.6, "Using the Page Hierarchy Security Editor."

If you want users to be able to edit pages at runtime, you can add Composer components to the pages.

For more information, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."

13.2 Adding Service Task Flows to Your Portal

WebCenter Portal services enrich portals and web sites with enterprise 2.0 capabilities, including social computing services, personal productivity services, online awareness and communications, content integration, and web analytics. WebCenter Portal services expose their functionality through task flows. For more information about WebCenter Portal services and task flows, see Section 2.4, "Introducing Oracle WebCenter Portal: Services."

If you create your portal using WebCenter Portal's Framework application template, all the appropriate WebCenter Portal service connection wizards and tag libraries are readily visible and available in the New Gallery and Component Palette. When you consume a WebCenter Portal service task flow or component, the necessary libraries are automatically added to the project. Depending on the service you plan to consume, your application must meet certain prerequisites. For example, if the service must know the identity of users, then your application must provide some level of security with user authentication. For more information, see Chapter 7, "Preparing Your Application for WebCenter Portal Services."

For more information about the individual WebCenter Portal services and their task flows, refer to Table 13–1.

Table 13–1 WebCenter Portal Services - More Information

WebCenter Portal Service	More Information
Activity Graph	Chapter 46, "Integrating the Activity Graph Service"
Analytics	Chapter 47, "Integrating the Analytics Service"
Announcements	Chapter 33, "Integrating the Announcements Service"
Blogs	Chapter 32, "Integrating Wikis and Blogs"
Discussions	Chapter 34, "Integrating the Discussions Service"
Documents	Chapter 30, "Integrating the Documents Service"
Events	Chapter 52, "Integrating the Events Service"
Instant Messaging and Presence (IMP)	Chapter 35, "Integrating the Instant Messaging and Presence Service"
Links	Chapter 43, "Integrating the Links Service"
Lists	Chapter 48, "Integrating the Lists Service"
Mail	Chapter 36, "Integrating the Mail Service"
Notifications	Chapter 49, "Integrating Notifications"
People Connections	Part VI, "Working with the People Connections Service"
Polls	Chapter 37, "Integrating the Polls Service"
Recent Activities	Chapter 50, "Integrating the Recent Activities Service"
RSS	Chapter 53, "Integrating the RSS Service"
Search	Chapter 45, "Integrating the Search Service"
Tags	Chapter 44, "Integrating the Tags Service"
Wikis	Chapter 32, "Integrating Wikis and Blogs"

Table 13–1 (Cont.) WebCenter Portal Services - More Information

WebCenter Portal Service	More Information
Worklist	Chapter 51, "Integrating the Worklist Service"

13.3 Adding Custom Task Flows to Your Portal

Task flows, like pages, are containers in which you can add components such as portlets, content, and other task flows. You can add task flows to pages or other task flows. Task flows created in JDeveloper can be included in Resource Catalogs so that users can add them to pages or other task flows at runtime.

You can drag and drop a custom task flow into a Resource Catalog so that it is available to users to add to pages or task flows at runtime. For more information, see Section 16.2.3.7, "Dragging and Dropping a Resource into the Catalog." When you deploy the application, the task flow is also deployed to the target instance.

For detailed information about creating different types of ADF task flows, see "Creating ADF Task Flows" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Exposing Task Flows Created in JDeveloper to a Deployed Application

If you created a custom task flow in JDeveloper and want to expose it in an already deployed application, you must deploy the project containing the task flow as an ADF shared library to the runtime application. Once the library containing the task flow is accessible to the application, it is displayed in the runtime Resource Registry and can be added to custom Resource Catalogs from there.

> **Note:** Permissions to view or edit a task flow are not provisioned by default when you create the task flow. To ensure that a task flow is visible at runtime, prior to deploying as a shared library, you must grant at least the View privilege on the task flow. For more information, see Section 24.6.1, "Granting Permissions on Task Flows."

For the steps to deploy an application to a Framework application, see "Deploying the Application to a WebLogic Managed Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. However, while running the Install Application Assistant, on the Choose targeting style page, select **Install this deployment as a library**.

For the steps to deploy an application to a WebCenter Portal: Spaces instance, see Section 56.2.3, "Rebuilding the Spaces Shared Library List."

13.4 Adding Content from Content Server to Your Portal

To provide access to the content in your Content Server content repository, you must create a connection to the Content Server. For more information, see Section 27.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter."

With a connection to Content Server established, you can use the Documents service, which includes task flows that provide a user-friendly interface to manage, display, and search documents at runtime. For more information, see Chapter 30, "Integrating the Documents Service."

Next, add one of the content task flows to your portal to expose the content from Content Server. Content task flows include Content Presenter and the Documents service task flows:

- Content Presenter
- Document Explorer
- Document List Viewer
- Document Manager
- Document Navigator
- Folder Viewer
- Recent Documents
- Document Viewer
- Document Mini Properties
- Document Properties
- Rich Text Editor
- Document Upload
- Document Version History

For more information about the Content Presenter task flow, see Section 31.1, "Understanding the Content Presenter Task Flow." For more information about the Documents service task flows, see Section 31.2, "Understanding the Documents Service Task Flows."

If you use the Content Presenter task flow, you may also want to create your own Content Presenter display templates. Content Presenter display templates define how content repository items should be rendered on a page. For more information, see Chapter 29, "Creating Content Presenter Display Templates."

13.5 Adding Portlets to Your Portal

A portlet is a reusable web component that can draw content from many different sources. Portlets provide a way of presenting data from multiple sources in a meaningful and related way. For more information about portlets, see Chapter 58, "Overview of Portlets."

To add a portlet to your portal, you must first register the portlet's producer with the Framework application. For more information, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."

After you have registered the portlet producer, you can then simply drag and drop any portlets that you want to include in your portal onto the appropriate page. For more information, see Section 64.3, "Adding Portlets to a Page."

For information about how to create portlets, see Chapter 60, "Creating Portlets with the Portlet Wizard."

13.6 Adding Pagelets to Your Portal

A pagelet is similar to a portlet, but while portlets were designed specifically for portals, pagelets can be run on any web page, including within a portal or other web application. Pagelets can be used to expose platform-specific portlets in other web environments.

For more information about adding pagelets to your portal, see Section 63.2, "Using Pagelets in Web Applications."

13.7 Adding External Applications to Your Portal

If the resources you add to your portal require access to applications that define their own authentication processes, you can add external applications to replicate a single sign-on experience for users.

For example, your portal might include portlets that point to applications that require a separate login. Or, if your portal includes the Mail service, you must identify an external application to map the mail server user to the portal user so that the user does not have to log in separately every time.

The first time a portal user accesses the external application, he or she is prompted for a user name and password for the application. These credentials are then mapped to the portal user and stored securely in a credential store. The credential store subsequently supplies these credentials during authentication to the external application. Unless the external application's credentials change, the user supplies the credentials only once as the mapped information is read from the credential store for future requests.

For more information, see Section 69.13, "Working with External Applications."

13.8 Adding Data Controls to Your Portal

Data controls are used to retrieve information from different data sources and create user interface (UI) components within your Framework application. A data control provides you with easy-to-use methods that you can drag and drop onto JSF pages to publish content as ADF components, such as URLs, files, and folders. To add content from the data sources to JSF pages, you must first create connections to the data sources, then use the connections to create data controls based on the repositories.

You can create custom data controls to connect different data sources and add them to your application pages or task flows. For detailed information about data controls, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Some WebCenter Portal services provide data controls for creating customized visualizations of the services. For information about consuming these data controls in your application, see Section 7.1.3, "Using WebCenter Portal Data Controls."

Exposing Data Controls for Consumption in an Already Deployed Application

The runtime Resource Registry in a Framework application or WebCenter Portal: Spaces contains a `Design Time Data Controls` folder by default. This folder exposes the seeded WebCenter Portal service data controls and any other data controls that were added to the application before it was deployed. To expose a new data control in an already deployed Framework application or WebCenter Portal: Spaces, you must deploy the project containing the data control as an ADF shared library to the runtime application. Once the library containing the data control is accessible to the application, it is displayed in the runtime Resource Registry and can be added to custom Resource Catalogs from there.

For the steps to deploy an application to a Framework application, see "Deploying Applications Using the WLS Administration Console" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. However, while running the Install

Application Assistant, on the Choose targeting style page, select **Install this deployment as a library**.

For the steps to deploy an application to a WebCenter Portal: Spaces instance, see Section 56.2.3, "Rebuilding the Spaces Shared Library List."

14

Visualizing Your Portal Navigation

This chapter shows you how to take the navigation model that has been defined for your WebCenter Portal: Framework application and expose it in the pages of the portal.

This chapter includes the following sections:

- Section 14.1, "Introduction to Navigation Visualization"
- Section 14.2, "Adding Built-in Navigation Task Flows to a Page Template"
- Section 14.3, "Programmatically Adding Navigation to a Page Template"
- Section 14.4, "Using Portal Navigation to Create a Sitemap."

For an introduction to navigation in WebCenter Portal: Framework applications, see Section 11, "Building a Navigation Model for Your Portal."

For information about how to define the navigation model for your portal, see Section 11.2, "Creating a Navigation Model" and Section 11.3, "Adding Resources to a Navigation Model."

14.1 Introduction to Navigation Visualization

The *navigation visualization* determines how the navigation appears in the portal. For example, navigation can be provided as a set of tabs along the top of each page, or perhaps as a tree structure along the side of the page. Another common navigation visualization is to provide a "master-detail" navigation, for example, tabs along the top of the page, with each tab having additional navigation provided as a tree structure along the side of the page.

Typically, you add navigation visualization to page templates, so that it can be defined in one place and propagated consistently across the whole portal. However, you can also add navigation visualization to individual pages.

WebCenter Portal provides several built-in navigation task flows that you can add to your portal's page templates to visualize the navigation. For more information, see Section 14.2, "Adding Built-in Navigation Task Flows to a Page Template."

The built-in navigation task flows provide a quick way to test your navigation model, but they provide a fairly simple navigation visualization. It is more likely that you will require a more advanced visualization. To do this, you must create your own navigation UI using ADF navigation components and the WebCenter Portal navigation APIs. For more information, see Section 14.3, "Programmatically Adding Navigation to a Page Template."

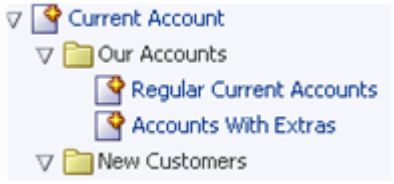
14.2 Adding Built-in Navigation Task Flows to a Page Template

You can add navigation to a page template by dragging one of the built-in navigation task flows from the Resource Palette onto the appropriate location of the template.

WebCenter Portal provides three navigation task flows to help visualize portal navigation. These are:

- **Navigation - Tree task flow**—Tree navigations display navigational links in a hierarchical structure and often contain multiple-level hierarchies (Figure 14–1). Tree navigations can provide relatively simple access to a complex structure. Typically, a tree-style navigation is displayed on the side of the page.

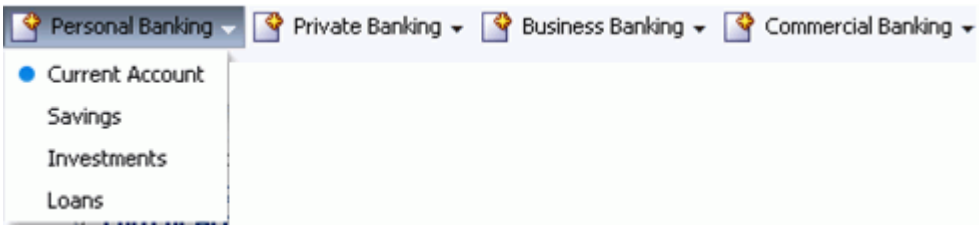
Figure 14–1 Navigation - Tree Task Flow



- **Navigation - Menu task flow**—Menu navigations display navigational links as a series of tabs, menus, bars, buttons, choices, or as a list, usually horizontally across the top or bottom of a page (Figure 14–2). Menu navigation is appropriate when linking to pages and documents, and where the number of items is known, such that there is enough room to display all the options that are available. Menu navigation is useful if you want to have a dropdown menu on the top of your application.

> **Note:** The Navigation Menu task flow can render only two levels of navigation.

Figure 14–2 Navigation - Menu Task Flow



- **Navigation - Breadcrumb task flow**—Breadcrumb navigations display a series of horizontal or vertical links that show users their current position in the navigation hierarchy (Figure 14–3). Breadcrumbs are particularly useful for deep hierarchies, say four levels or more, to enable users to easily navigate back to a specific place in the hierarchy.

Figure 14–3 Navigation - Breadcrumb Task Flow



Additional navigation task flows are available to download from Oracle Technology Network, at:

http://www.oracle.com/technetwork/middleware/webcenter/samples-196325.zip

The zip file contains the following navigation task flows:

- List Navigation—Renders a navigation model in a vertical list format. Folders and their items are rendered as dropdown menus. Empty folders are not rendered.
- Menu with Links Navigation—Renders a navigation model in a horizontal menu format. For top level navigation items, folders are rendered as dropdown menus, navigable links with no children are rendered as a top level link. Empty folders are not rendered.
- Tree Navigation—Renders a navigation model in a vertical collapsed tree format.

Download the zip file and extract its contents locally. You can then upload one or more of the EAR files to your application, as described in Section 17.4.4, "How to Upload a Resource Using the Resource Manager."

This section includes the following subsections:

- Section 14.2.1, "How to Add a Navigation Task Flow to a Page Template"
- Section 14.2.2, "Navigation Task Flow Parameters"
- Section 14.2.3, "What You May Need to Know About Navigation Task Flow Style Classes"

14.2.1 How to Add a Navigation Task Flow to a Page Template

The steps for adding a navigation task flow are the same for all navigation task flows.

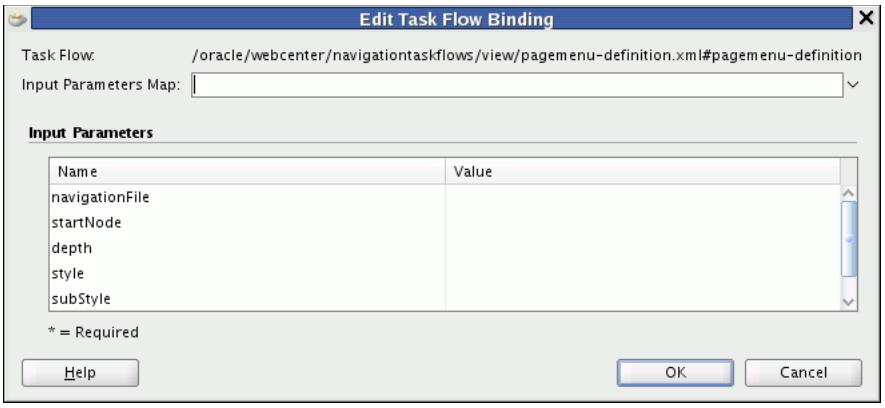
To add a navigation task flow:

1. Open the page template to which you want to add the navigation task flow.
2. In the Resource Palette, expand the **WebCenter Portal - Services Catalog**. Under **Task Flows**, select the desired navigation task flow and drag it onto the desired location of the template.
3. From the menu that displays, choose **Region** to open the Edit Task Flow Binding dialog.

   You may be prompted to add the Navigation Task Flows library to the project. Confirm by clicking **Add Library**.
4. In the Edit Task Flow Binding dialog (Figure 14–4), if necessary, specify parameter values for the task flow. For more information, see Section 14.2.2, "Navigation Task Flow Parameters."

> **Note:** You can use EL expressions as values of any of the navigation task flow binding parameters. For more information, see Appendix G, "Expression Language Expressions."

Figure 14–4 The Edit Task Flow Binding dialog for the Navigation - Menu Task Flow



5. Click **OK**.

14.2.2 Navigation Task Flow Parameters

Table 14–1 lists and describes task flow binding parameters applicable to the Navigation - Tree task flow.

Note: You can use EL expressions for the values of any of the navigation task flow binding parameters. For more information, see Appendix G, "Expression Language Expressions."

Table 14–1 Navigation - Tree Task Flow Binding Parameters

Parameter	Default Value	Description
navigationFile	/oracle/webcenter/portalapp/navigations/default-navigation-model	The navigation model to associate with this task flow. You can provide the actual navigation model definition file, omitting the .xml, for example: /oracle/webcenter/portalapp/navigations/myNavigationModel You can also enter an EL expression, for example: #{navigationContext.defaultNavigationModel}
startNode	/	The point in the navigation model at which to start the navigation. For example: ■ To use a specific page as the starting point, or root, of the navigation, enter the path to the page, for example: pages_home ■ To use the currently selected page as the root of the navigation, use the following EL expression: #{navigationContext.navigationModel['modelPath=*path*'].currentSelection.prettyUrl} ■ To use a particular level of the path to the currently selected page as the root, use the following EL expression: #{navigationContext.navigationModel['modelPath=*path*'].currentSelection.prettyUrlPath[*level*]} where: *path* is the path to the navigation model XML file (omitting the trailing .xml), for example: /oracle/webcenter/portalapp/navigations/myNavigationModel *level* is the level in the path at which to start.
showRoot	none	Specifies whether to render the starting point of the navigation (root) in the tree. Enter one of the following: ■ none—Do not render the root ■ node—Render the root ■ box—Render the root as the header in a box

Table 14–1 (Cont.) Navigation - Tree Task Flow Binding Parameters

Parameter	Default Value	Description
`depth`	`1`	The number of levels shown when the tree is initially rendered. Enter one of the following: ■ `1`—Show top level nodes only; all other notes are initially collapsed ■ `2`—Expand the first level of child nodes ■ `3`—Expand the first and second level of child nodes ■ `all`—Expand all nodes **Note:** This parameter applies only if the `style` parameter is set to `collapsed`. If the `style` parameter is set to `expanded`, then all nodes are shown regardless of the value of this parameter.
`style`	`collapsed`	The style of the navigation. Enter one of the following: ■ `collapsed`—Render the given levels (as specified by the `depth` parameter) as initially collapsed ■ `expanded`—Render all tree nodes, expanded to show all children ■ `list`—Render a flat list of top level resources. If any of the top-level resources are folders, the content of those folders is provided in dropdown menus. **Note:** Entering `list` overrides the `depth` and `showRoot` parameters.
`showIcons`	`true`	Specifies whether to render icons for each resource. Enter `true` or `false`.
`navigationEditMode`	Not used	Not used

Table 14–2 lists and describes task flow binding parameters applicable to the Navigation - Menu task flow.

Table 14–2 Navigation - Menu Task Flow Binding Parameters

Parameter	Default Value	Description

`navigationFile`	`/oracle/webcenter/portalapp/navigations/default-navigation-model`	The navigation model to associate with this task flow. You can provide the actual navigation model definition file, omitting the `.xml`, for example:
		`/oracle/webcenter/portalapp/navigations/myNavigationModel`
		You can also enter an EL expression, for example:
		`#{navigationContext.defaultNavigationModel}`

`startNode`	`/`	The point in the navigation model at which to start the navigation.
		For example:
		■ To use a specific page as the starting point, or root, of the navigation, enter the path to the page, for example:
		`pages_home`
		■ To use the currently selected page as the root of the navigation, use the following EL expression:
		`#{navigationContext.navigationModel['modelPath=`*`path`*`'].currentSelection.prettyUrl}`
		■ To use a particular level of the path to the currently selected page as the root, use the following EL expression:
		`#{navigationContext.navigationModel['modelPath=`*`path`*`'].currentSelection.prettyUrlPath[`*`level`*`]}`
		where:
		`path` is the path to the navigation model XML file (omitting the trailing `.xml`), for example:
		`/oracle/webcenter/portalapp/navigations/myNavigationModel`
		`level` is the level in the path at which to start.

`depth`	Not used	Not used

Table 14–2 (Cont.) Navigation - Menu Task Flow Binding Parameters

Parameter	Default Value	Description
`style`	`menu`	The style for rendering the first level of resources. Enter one of the following: `tabs`, `bar`, `buttons`, `choice`, `list`, `menu` **Notes:** ■ If the resource being rendered is non-navigable, for example, a folder, then it cannot be displayed as a tab, bar, button, choice, or list. It can be displayed only if the `style` parameter is set to `menu`. ■ If `style` is set to `menu`, then a folder is rendered with one level of dropdown menu. ■ If `style` is set to `menu` and `subStyle` is set to `menu`, then a folder is rendered with two levels of dropdown menu. ■ If a navigable item is rendered with `style` and `subStyle` set to `menu`, the item is rendered twice, once as a dropdown menu, and once as a link to the item itself.
`subStyle`	`menu`	The style for rendering the second level of navigation items if applicable. If `style` is set to `menu`, enter `menu` or `donotdisplay`. If `style` is set to any other value, enter `tabs`, `bar`, `buttons`, `choice`, `list`, or `donotdisplay`.
`showIcons`	`true`	Specifies whether to render icons for each resource. Enter `true` or `false`.

Table 14–3 lists and describes task flow binding parameters applicable to the Navigation - Breadcrumb task flow.

Table 14–3 Navigation - Breadcrumb Task Flow Binding Parameters

Parameter	Default Value	Description
navigationFile	/oracle/webcenter/portalapp/navigations/default-navigation-model	The navigation model to associate with this task flow. You can provide the actual navigation model definition file, omitting the .xml, for example: /oracle/webcenter/portalapp/navigations/myNavigationModel You can also enter an EL expression, for example: #{navigationContext.defaultNavigationModel}
startNode	/	The point in the navigation model at which to start the navigation. For example: ■ To use a specific page as the starting point, or root, of the navigation, enter the path to the page, for example: pages_home ■ To use the currently selected page as the root of the navigation, use the following EL expression: #{navigationContext.navigationModel['modelPath=*path*'].currentSelection.prettyUrl} ■ To use a particular level of the path to the currently selected page as the root, use the following EL expression: #{navigationContext.navigationModel['modelPath=*path*'].currentSelection.prettyUrlPath[*level*]} where: *path* is the path to the navigation model XML file (omitting the trailing .xml), for example: /oracle/webcenter/portalapp/navigations/myNavigationModel *level* is the level in the path at which to start.
showRoot	false	Specifies whether to show the start node (root) in the breadcrumbs. Enter true or false.
style	horizontal	Specifies whether to display breadcrumbs horizontally or vertically. Enter horizontal or vertical.

14.2.3 What You May Need to Know About Navigation Task Flow Style Classes

WebCenter Portal provides a style class for each of the navigation task flows that you can use to change the appearance of the task flows. To use these style classes, add them to the skin CSS file used by the application, for example portal-skin.css.

- WCTreeNav—Use to change the appearance of the Navigation - Tree task flow
- WCMenuNav—Use to change the appearance of the Navigation - Menu task flow
- WCBreadcrumbNav—Use to change the appearance of the Navigation - Breadcrumb task flow.

Example 14–1 shows what to add to the skin CSS file to change the background color for the Navigation - Menu task flow.

Example 14–1 Changing the Background Color of the Navigation - Menu Task Flow

```
.WCMenuNav af|menuBar {
  background-color: red;
}
```

For more information about skins, see Chapter 15, "Creating and Managing Skins."

14.3 Programmatically Adding Navigation to a Page Template

If the built-in navigation task flows do not meet your requirements, you can use the APIs provided with WebCenter Portal to create your own UI for your navigation model.

Note: Any task flow that uses the navigation model to trigger navigation within an application must include a `parent-action` activity, named `wcnav_parentAction`, in the task flow definition that propagates the `wcnav_outcome` to the root level, as follows:

```
<parent-action id="wcnav_parentAction">
  <root-outcome>wcnav_outcome</root-outcome>
</parent-action>
```

WebCenter Portal provides two sets of APIs for adding navigation to your portal:

- Expression Language (EL) APIs—Use EL expressions to obtain the navigation model and navigation resources.
- REST APIs—Use standard HTTP methods to point to the navigation model and navigation resources, returning a standard representation of any retrieved object. REST APIs can be used when using client-side programming languages, for example JavaScript + HTML, or environments, for example an iPhone.

This section includes the following subsections:

- Section 14.3.1, "Using the Navigation Expression Language APIs"
- Section 14.3.2, "Using the Navigation REST APIs"

14.3.1 Using the Navigation Expression Language APIs

WebCenter Portal provides a set of expression language (EL) APIs that you can use to obtain the navigation model and represent that navigation model as a runtime model. The runtime models can be bound directly to ADF Faces navigation components.

The available navigation EL expressions are listed in full in Section G.5, "ELs Related to Navigation."

This section includes the following subsections:

- Section 14.3.1.1, "Introduction to the Navigation Context"
- Section 14.3.1.2, "Introduction to the Navigation Runtime Model"
- Section 14.3.1.3, "Introduction to the Navigation Resource"
- Section 14.3.1.4, "How to Render the Navigation Model as a List of Links"
- Section 14.3.1.5, "How to Render the Navigation Model as a Tree"
- Section 14.3.1.6, "How to Render the Navigation Model as a Three-Level Menu"

- Section 14.3.1.7, "How to Render the Navigation Model as Breadcrumbs"
- Section 14.3.1.8, "How to Render Master-Detail Navigation"

> **Note:** This section includes some examples of how to use the navigation EL APIs. For a full working example, see the default page templates (`pageTemplate_globe.jspx` and `pageTemplate_swooshy.jspx`) created when you create an application using the WebCenter Portal: Framework Application template.

14.3.1.1 Introduction to the Navigation Context

The Navigation Context is the entry point to access all navigation elements within your system. Specifically, it enables you to access navigation models via preference, state, and direct reference, as well as providing the hooks to create custom UI to actually navigate to the resource.

Navigation Model Access

- Default navigation model (preference)—This is the model specified by the ADF preference `oracle.webcenter.portalapp.navigation.model` in the `adf-config.xml` file. This enables you to define the general model to be used throughout your application and enables you to change the value in one place rather than having to go to each page and page template to set the value. The EL expression to retrieve the default navigation model is:

  ```
  #{navigationContext.defaultNavigationModel}
  ```

  > **Note:** For more information about how to set the `oracle.webcenter.portalapp.navigation.model` preference, see Section 11.4, "Selecting the Default Navigation Model."

- Current navigation model (state)—This model is the one used to navigate to the current view and, if it is not set, returns the default navigation model. Your application may contain multiple navigation models. The EL expression to retrieve the current navigation model is:

  ```
  #{navigationContext.currentNavigationModel}
  ```

  Using this expression enables you to have a single value within a page template (for example, for displaying breadcrumbs) and have it display the correct value without having to select a specific navigation model.

- Direct navigation model access (direct reference)—You can also select a specific navigation model by passing in the path to the XML definition of the model. The EL expression to retrieve a specific navigation model is:

  ```
  #{navigationContext.navigationModel['modelPath=path']
  ```

  where *`path`* is the path to the navigation model definition file, for example:

  ```
  #{navigationContext.navigationModel['modelPath=/oracle/webcenter/portalapp/navigations/myNavigation']}
  ```

  > **Note:** You do not need to include the `.xml` file name extension.

Resource Navigation

Use these ELs for core bean operations for binding the navigation model to the UI component's `actionListener` attribute. For example:

```
#{navigationContext.processAction}
```

Examples

Example 14–2 creates a simple tree UI using the Navigation Context to access the current navigation model and render the tree using `<af:outputText/>`.

Example 14–2 Simple Tree Navigation UI

```
<af:tree var="node"
    value="#{navigationContext.currentNavigationModel.treeModel['startNode=/,
    includeStartNode=true,
    depth=1']}"
    id="t1">
  <f:facet name="nodeStamp">
    <af:outputText value="#{node.title}" id="ot1"/>
  </f:facet>
</af:tree>
```

Example 14–3 uses the Navigation Context's `processAction` listener to handle navigation for an ADF `commandImageLink` component, passing in the current node as the parameter called *node* as an attribute to the `actionListener`.

Example 14–3 Binding the Navigation Model to the UI Component

```
<af:tree var="node"
    value="#{navigationContext.currentNavigationModel.treeModel['startNode=/,
           includeStartNode=true,
           depth=1']}"
    id="t1">
  <f:facet name="nodeStamp">
    <af:commandImageLink text="#{node.title}"
        id="sm_c1b"
        actionListener="#{navigationContext.processAction}"
        inlineStyle="#{node.selected ? 'font-weight:bold;' : ''}">
      <f:attribute name="node" value="#{node}"/>
    </af:commandImageLink>
  </f:facet>
</af:tree>
```

14.3.1.2 Introduction to the Navigation Runtime Model

The Navigation Runtime Model exposes the underlying XML definition as runtime models, as well as providing direct access to individual nodes. The runtime models take into account such factors as security and visibility to produce a session-specific representation that can be bound to UI objects.

Note: Note the difference between the *navigation model* that you create to define the content, structure, and metadata of the navigation and the *navigation **runtime** model* that determines how the navigation model behaves at runtime.

You can access the whole model by using the default runtime model (`defaultTreeModel`, `defaultListModel`, `defaultMenuModel`, and `defaultSiteMap`), or access a specific sub-tree using the various parameters when creating the runtime model. For example:

```
#{navigationContext.defaultNavigationModel.menuModel['startNode=home,
    includeStartNode=false, depth=2']}
```

For more information about the underlying ADF Faces MenuModel, TreeModel, and ListModel, see the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

Model Access

You can create the following runtime models based on the underlying navigation model:

- UI Models
  - Tree model

    ```
    #{navigationContext.defaultNavigationModel.defaultTreeModel}
    #{navigationContext.defaultNavigationModel.treeModel['parameters']}
    ```

  - Menu model

    ```
    #{navigationContext.defaultNavigationModel.defaultMenuModel}
    #{navigationContext.defaultNavigationModel.menuModel['parameters']}
    ```

  - List model

    ```
    #{navigationContext.defaultNavigationModel.defaultListModel}
    #{navigationContext.defaultNavigationModel.listModel['parameters']}
    ```

- Search Engine Model
  - Sitemap

    ```
    #{navigationContext.defaultNavigationModel.defaultSiteMap}
    #{navigationContext.defaultNavigationModel.siteMap['parameters']}
    ```

Resource (or Node) Access

You can access specific nodes within the navigation model using the follow EL expressions:

- `#{navigationContext.defaultNavigationModel.currentSelection}`
- `#{navigationContext.defaultNavigationModel.rootNode}`
- `#{navigationContext.defaultNavigationModel.node['path']}`

Examples

Example 14–4 renders the current navigation model's menu model as a breadcrumb.

Example 14–4 Rendering the Navigation Model as a Menu Model

```
<af:breadCrumbs id="bc1"
    var="node"
    value="#{navigationContext.currentNavigationModel.defaultMenuModel}">
  <f:facet name="nodeStamp">
    <af:commandNavigationItem id="cni1"
        text="#{node.title}"
```

```
        actionListener="#{navigationContext.processAction}"
        partialSubmit="true">
      <f:attribute name="node" value="#{node}"/>
    </af:commandNavigationItem>
  </f:facet>
</af:breadCrumbs>
```

Example 14–5 produces a Sitemap for the application based on the default navigation model.

Example 14–5 Producing a Sitemap

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"  version="2.1"
    xmlns:f="http://java.sun.com/jsf/core">
  <jsp:directive.page contentType="text/xml;charset=UTF-8" />
  <f:view>
    ${navigationContext.defaultNavigationModel.defaultSiteMap}
  </f:view>
</jsp:root>
```

For more information about Sitemaps, see Section 14.4, "Using Portal Navigation to Create a Sitemap."

14.3.1.3 Introduction to the Navigation Resource

The Navigation Resource provides access to individual properties against each node within the navigation model. These fall into the following categories:

- **Attributes**—Common properties that the user can specify against any navigation element and are used for rendering the node on the page, including:
  - **Title**—The title displayed for the node when the navigation model is rendered on a page.
  - **AccessKey**—A key mnemonic (single character) that users can enter to access the node without using the mouse.
  - **Description**—A description of the node.
  - **IconURI**—An icon to visually represent the node. This is displayed next to the Title when the navigation model is rendered on a page.
  - **Subject**—Keywords to facilitate searching of the node.
  - **Target**—The location on the container page where the node is displayed when it is selected, either in the same browser window (`_self`), a new window (`_blank`), or a popup (`_popup`), or any other location supported by the navigation UI.
  - **ToolTip**—Text that displays to provide additional information about the node when users hover the mouse over the Title.
  - **Modified**—The date of the last modification of the node. This attribute is used for Sitemap creation.
  - **ChangeFrequency**—How frequently the node is likely to change: always, hourly, daily, weekly, monthly, yearly, never. This attribute is used for Sitemap creation.
  - **Significance**—The priority of this resource relative to other nodes in the navigation model, within the range 0.0 to 1.0. This attribute is used for Sitemap creation.

  - **ExternalId**—An ID to enable a direct reference to any node in the navigation model from a static link in the page.

- **Parameters**—User-defined properties that are specific to each node. These are name/value pairs and can contain any value.
- **State**—Built-in properties that you can query and use to navigate to the node. For example:
  - Type of node, for example, folder or separator
  - Whether the node is navigable
  - The path to the node
  - Parent, child, or sibling node access
  - Whether the node is the currently selected node within the navigation model or on the selected node's path

  For a complete list, see Section G.5, "ELs Related to Navigation."

Examples

Example 14–6 renders an ADF `commandImageLink` component with various attributes.

Example 14–6 Rendering a commandImageLink with Attributes

```
<af:commandImageLink text="#{node.title}"
    id="cil1"
    actionListener="#{navigationContext.processAction}"
    shortDesc="#{node.attributes['ToolTip']}"
    accessKey="#{node.attributes['AccessKey']}"
    inlineStyle="#{node.selected ? 'font-weight:bold;' : ''}">
  <f:attribute name="node" value="#{node}"/>
</af:commandImageLink>
```

Example 14–7 accesses values passed from a resource to the corresponding page through parameters.

Example 14–7 Passing Node Values to Page Parameters

homePageDef.xml

```
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
    version="11.1.1.55.96" id="homePageDef"
    Package="oracle.webcenter.portalapp.pages">
  <parameters>
    <parameter id="NavigationParameter"
        value="#{navigationContext.currentNavigationModel.currentSelection.parameters
            ['MyNavParam']}"/>
  </parameters>
```

Example 14–8 conditionally displays breadcrumb links based on whether you can navigate to the node. It also bases the UI component on the type of the node.

Example 14–8 Conditionally Displaying Navigation Nodes

```
<af:breadCrumbs id="bc1"
                var="node"
                value="#{navigationContext.currentNavigationModel.defaultMenuModel}">
  <f:facet name="nodeStamp">
```

```
    <af:switcher facetName="#{node.navigable}" id="swn1">
      <f:facet name="true">
        <af:commandNavigationItem id="cni1" text="#{node.title}"
                                  actionListener="#{navigationContext.processAction}"
                                  partialSubmit="true">
          <f:attribute name="node" value="#{node}"/>
        </af:commandNavigationItem>
      </f:facet>
      <f:facet name="false">
        <af:outputText value="#{node.title}" rendered="#{!node.separator}" id="ot3"/>
      </f:facet>
    </af:switcher>
  </f:facet>
</af:breadCrumbs>
```

14.3.1.4 How to Render the Navigation Model as a List of Links

One common way to visualize a portal's navigation is as a list of links. Clicking a link navigates to the resource associated with it.

Example 14–9 retrieves the list model for the current navigation model and renders it vertically on the page. The code iterates through the navigation model, binding each node in turn to an ADF `commandImageLink` component. If a node contains children, a second iterator renders those children on the page. If a node is not navigable, it is not rendered as a link. The currently selected node is displayed in bold.

Example 14–9 Navigation as a List of Links

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp1" height="20px"/>
  <af:iterator id="i1"
               value="#{navigationContext.currentNavigationModel.listModel['startNode=/,
                     includeStartNode=false']}"
               var="node">
    <af:panelGroupLayout layout="vertical">
      <af:commandImageLink id="cil2" text="#{node.title}"
                           actionListener="#{navigationContext.processAction}"
                           action="pprnav"
                           icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                           disabled="#{not node.navigable}"
                           inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">
        <f:attribute name="node" value="#{node}"/>
      </af:commandImageLink>
      <af:iterator id="i2" value="#{node.children}" var="node2">
        <af:panelList id="pl1">
          <af:commandImageLink id="cil3" text="#{node2.title}"
                               actionListener="#{navigationContext.processAction}"
                               action="pprnav"
                               icon="#{node2.attributes[pageFlowScope.tnBean.iconKey]}"
                               disabled="#{not node2.navigable}"
                               inlineStyle="#{node2.onSelectedPath ? 'font-weight:bold;' : ''}">
            <f:attribute name="node" value="#{node2}"/>
          </af:commandImageLink>
        </af:panelList>
      </af:iterator>
    </af:panelGroupLayout>
  </af:iterator>
  <af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

14.3.1.5 How to Render the Navigation Model as a Tree

Another common way to visualize portal navigation is as a tree structure. A navigation tree is typically listed down the side of the page, so it is a useful way of displaying the entire navigation hierarchy. Users can navigate immediately to any node in the navigation. The advantage of using a tree structure over a simple list of links, is that users can expand and collapse the different nodes of the navigation model.

To include a tree structure in your page template, you can bind the ADF Faces `tree` navigation component to your navigation model. Example 14–10 shows how to do this.

Example 14–10 Navigation as a Tree

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp2" height="20px"/>
  <af:tree id="tree1" var="node" initiallyExpanded="true"
           value="#{navigationContext.currentNavigationModel.treeModel['includeStartNode=false']}">
    <f:facet name="nodeStamp">
      <af:commandImageLink id="cil2" text="#{node.title}"
                           actionListener="#{navigationContext.processAction}"
                           action="pprnav"
                           icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                           disabled="#{not node.navigable}"
                           inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">
        <f:attribute name="node" value="#{node}"/>
      </af:commandImageLink>
    </f:facet>
  </af:tree>
  <af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

14.3.1.6 How to Render the Navigation Model as a Three-Level Menu

You can also render your navigation as a menu. In a menu, the top level of navigation is displayed. When the user hovers the mouse over a particular node, if that node has children, they are displayed as a popup menu.

Figure 14–11 retrieves the list model for the current navigation model and renders it vertically on the page. The code iterates through the navigation model. If a node contains children, then a second iterator iterates through those children. Nodes without children are rendered as menu items. In the second level of the navigation model, if any of the nodes have children, then a third iterator is executed. Again, nodes without children are rendered as menu items. At the third level of the navigation mode, nodes are rendered as menu items. Non-navigable nodes are not rendered as links. The currently selected node is displayed in bold.

Example 14–11 Navigation as a Menu

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp1" height="20px"/>
  <af:menuBar id="mb1">
    <af:iterator id="i1"
                 value="#{navigationContext.currentNavigationModel.listModel['startNode=/,
                       includeStartNode=false']}"
                 var="node">
      <af:switcher id="s1"
                   facetName="#{empty node.children ? 'leafNode' : 'parentNode'}">
        <f:facet name="parentNode">
          <af:menu id="m1" text="#{node.title}"
```

```
              inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">
          <af:iterator id="i2" value="#{node.children}"
                       var="node2">
            <af:switcher id="s2"
                         facetName="#{empty node2.children ? 'leafNode' : 'parentNode'}">
              <f:facet name="parentNode">
                <af:menu id="m2" text="#{node2.title}"
                         inlineStyle="#{node2.onSelectedPath ? 'font-weight:bold;' : ''}">
                  <af:iterator id="i3" value="#{node2.children}"
                               var="node3">
                    <af:commandMenuItem id="cml3"
                                        text="#{node3.title}"
                                        actionListener="#{navigationContext.processAction}"
                                        action="pprnav"
                                        icon="#{node3.attributes[pageFlowScope.tnBean.iconKey]}"
                                        disabled="#{not node3.navigable}"
                                        inlineStyle="#{node3.onSelectedPath ?
                                                       'font-weight:bold;' : ''}">
                      <f:attribute name="node" value="#{node3}"/>
                    </af:commandMenuItem>
                  </af:iterator>
                </af:menu>
              </f:facet>
              <f:facet name="leafNode">
                <af:commandMenuItem id="cml1"
                                    text="#{node2.title}"
                                    actionListener="#{navigationContext.processAction}"
                                    action="pprnav"
                                    icon="#{node2.attributes[pageFlowScope.tnBean.iconKey]}"
                                    disabled="#{not node2.navigable}"
                                    inlineStyle="#{node2.onSelectedPath ?
                                                   'font-weight:bold;' : ''}">
                  <f:attribute name="node" value="#{node2}"/>
                </af:commandMenuItem>
              </f:facet>
            </af:switcher>
          </af:iterator>
        </af:menu>
      </f:facet>
      <f:facet name="leafNode">
        <af:commandMenuItem id="cml2" text="#{node.title}"
                            actionListener="#{navigationContext.processAction}"
                            action="pprnav"
                            icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                            disabled="#{not node.navigable}"
                            inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">
          <f:attribute name="node" value="#{node}"/>
        </af:commandMenuItem>
      </f:facet>
    </af:switcher>
  </af:iterator>
 </af:menuBar>
 <af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

14.3.1.7 How to Render the Navigation Model as Breadcrumbs

To provide users with a quick way of orientating themselves within the portal navigation, you can provide a trail of breadcrumbs on the page. Breadcrumbs show the current location in the portal and the path taken through the navigation to get

there. Users can then quickly return to any point along that path. Breadcrumbs are typically used on a page in addition to other forms of navigation.

To include breadcrumbs in your page template, you can bind the ADF Faces `breadCrumbs` navigation component to your navigation model. Example 14–12 shows how to do this.

Example 14–12 Navigation as Breadcrumbs

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp1" height="20px"/>
  <af:breadCrumbs id="bc1" var="node"
                  value="#{navigationContext.currentNavigationModel.menuModel[
                        'includeStartNode=false']}">
    <f:facet name="nodeStamp">
      <af:commandNavigationItem id="cil1" text="#{node.title}"
                                actionListener="#{navigationContext.processAction}"
                                action="pprnav"
                                icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                                disabled="#{not node.navigable}">
        <f:attribute name="node" value="#{node}"/>
      </af:commandNavigationItem>
    </f:facet>
  </af:breadCrumbs>
  <af:spacer id="sp2" height="20px"/>
</af:panelGroupLayout>
```

14.3.1.8 How to Render Master-Detail Navigation

Often portals provide a "master-detail" navigation visualization, for example, as tabs along the top of the page, with each tab having additional navigation provided as a tree structure along the side of the page.

The master-detail navigation in Example 14–13 uses the same navigation model for the master and detail navigation. The master navigation renders the top level of the navigation model as a list of links along the top of the page. The detail navigation renders the children of the currently selected top level node as a tree.

Example 14–13 Master-Detail Navigation Using a Single Navigation Model

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp1" height="20px"/>
  <!-- Master -->
  <af:navigationPane id="np1" var="node" hint="bar" level="1"
                     value="#{navigationContext.navigationModel[
                           'modelPath=/oracle/webcenter/portalapp/navigations/master-detail.xml']
                           .defaultMenuModel}">
    <f:facet name="nodeStamp">
      <af:commandNavigationItem id="cil1" text="#{node.title}"
                                actionListener="#{navigationContext.processAction}"
                                action="pprnav"
                                icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                                disabled="#{not node.navigable}"
                                inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">

        <f:attribute name="node" value="#{node}"/>
      </af:commandNavigationItem>
    </f:facet>
  </af:navigationPane>
  <af:spacer id="sp2" height="20px"/>
  <!-- Setup the parameters for detail query -->
```

```
  <c:set value="startNode=/${navigationContext.navigationModel[
              'modelPath=/oracle/webcenter/portalapp/navigations/master-detail.xml']
              .currentSelection.prettyUrlPath[1]}, includeStartNode=false"
         var="currSel" scope="session"/>
  <!-- Detail -->
  <af:tree id="tree1" var="node2" initiallyExpanded="true"
           value="#{navigationContext.navigationModel[
                 'modelPath=/oracle/webcenter/portalapp/navigations/master-detail.xml']
                 .treeModel[currSel]}">
    <f:facet name="nodeStamp">
      <af:commandImageLink id="cil2" text="#{node2.title}"
                           actionListener="#{navigationContext.processAction}"
                           action="pprnav"
                           icon="#{node2.attributes[pageFlowScope.tnBean.iconKey]}"
                           disabled="#{not node2.navigable}"
                           inlineStyle="#{node2.onSelectedPath ? 'font-weight:bold;' : ''}">
        <f:attribute name="node" value="#{node2}"/>
      </af:commandImageLink>
    </f:facet>
  </af:tree>
  <af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

The example above can also be written to use two different navigation models, one for the master navigation, and one for the detail navigation. Example 14–14 shows how.

Example 14–14 Master-Detail Navigation Using Multiple Navigation Models

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:spacer id="sp1" height="20px"/>
  <!-- Master -->
  <af:navigationPane id="np1" var="node" hint="bar" level="1"
                     value="#{navigationContext.navigationModel[
                           'modelPath=/oracle/webcenter/portalapp/navigations/master.xml']
                           .defaultMenuModel}">
    <f:facet name="nodeStamp">
      <af:commandNavigationItem id="cil1" text="#{node.title}"
                                actionListener="#{navigationContext.processAction}"
                                action="pprnav"
                                icon="#{node.attributes[pageFlowScope.tnBean.iconKey]}"
                                disabled="#{not node.navigable}"
                                inlineStyle="#{node.onSelectedPath ? 'font-weight:bold;' : ''}">
        <f:attribute name="node" value="#{node}"/>
      </af:commandNavigationItem>
    </f:facet>
  </af:navigationPane>
  <af:spacer id="sp2" height="20px"/>
  <!-- Setup the parameters for detail query -->
  <c:set value="startNode=/${navigationContext.navigationModel[
              'modelPath=/oracle/webcenter/portalapp/navigations/master.xml']
              .currentSelection.prettyUrlPath[1]}, includeStartNode=false"
         var="currSel" scope="session"/>
  <!-- Detail -->
  <af:tree id="tree1" var="node2" initiallyExpanded="true"
                      value="#{navigationContext.navigationModel[
                            'modelPath=/oracle/webcenter/portalapp/navigations/master.xml']
                            .treeModel[currSel]}">
    <f:facet name="nodeStamp">
      <af:commandImageLink id="cil2" text="#{node2.title}"
                           actionListener="#{navigationContext.processAction}"
```

```
                          action="pprnav"
                          icon="#{node2.attributes[pageFlowScope.tnBean.iconKey]}"
                          disabled="#{not node2.navigable}"
                          inlineStyle="#{node2.onSelectedPath ? 'font-weight:bold;' : ''}">
        <f:attribute name="node" value="#{node2}"/>
      </af:commandImageLink>
    </f:facet>
  </af:tree>
  <af:spacer id="sp3" height="20px"/>
</af:panelGroupLayout>
```

14.3.2 Using the Navigation REST APIs

Oracle WebCenter Portal provides REST APIs to access and manipulate the navigation model of your application. Use the navigation REST APIs to create your own interface for visualizing the navigation model. You would typically use REST when creating Rich Internet Applications that are client-side scripted and require the ability to interact with data from a server-side application. For example, the WebCenter Portal iPhone App uses WebCenter Portal REST APIs to interact with a WebCenter Portal: Spaces application.

For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

The starting point for using the navigation REST APIs, as with other WebCenter Portal REST APIs, is to send a `GET` request to the Resource Index:

```
http://host:port/rest/api/resourceIndex
```

The response to this request provides the entry point URI (`href`) for the navigation REST APIs, identified by the link element with the following `resourceType`:

```
urn:oracle:webcenter:navigations
```

Sending a `GET` request to this entry point URI returns a list of navigation models within the application, for example:

```
<sitestructure>
  ...
  <navigation-list>
    oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/
    navigations/default-navigation
  </navigation-list>
  <navigation-list>
    oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/
    navigations/admin-navigation
  </navigation-list>
  <navigation-list>
    oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/
    navigations/myNavigation
  </navigation-list>
  ...
</sitestructure>
```

Using this information, you can start to retrieve information about specific navigation models and the resources within them, using the URIs listed in Table 14–4.

Table 14–4 REST URI Model for Navigation

REST URI	Returns
`.../api/navigations`	All navigation models in the application.
`.../api/navigations/default-navigation-model`	The properties of the default navigation model.
`.../api/navigations/`*`pathToModel`*`/`*`modelId`*	The properties of the specified navigation model (*`modelId`*).
`.../api/navigations/`*`pathToModel`*`/`*`modelId`*`/resources`	All root resources under the specified navigation model (*`modelId`*).
`.../api/navigations/`*`pathToModel`*`/`*`modelId`*`/resources/`*`pathToResource`*`/`*`resourceId`*	The specified resource (*`resourceID`*) from the specified navigation model (*`modelId`*).
`...api/navigations/`*`pathToModel`*`/`*`modelId`*`/resources/`*`pathToResource`*`/`*`resourceId`*`/children`	A navigation tree listing all the children of the specified resource (*`resourceId`*) from the specified navigation model (*`modelId`*).
`.../api/navigations/`*`pathToModel`*`/`*`modelId`*`/resources/`*`pathToResource`*`/`*`resourceId`*`/children?depth=`*`dep`*	A navigation tree listing the children of the specified resource (*`resourceId`*) from the specified navigation model (*`modelId`*) up to a specified level (*`dep`*).

Each navigable resource has a `prettyURL` element that you can use to drill down further in the navigation model hierarchy.

14.4 Using Portal Navigation to Create a Sitemap

When a search engine crawls a web site, it parses the HTML for standard links (for example, `<a href="url">text</a>`). However, ADF-based web sites generate links as JavaScript, which when executed in a browser, cause the page change or navigation. These JavaScript links cannot be interpreted by search engines.

Some popular search engines, such as Google and Bing, support Sitemaps. A Sitemap is an XML file that lists URLs for a site along with additional metadata about each URL (for example, when it was last updated, how often it usually changes, and how important it is relative to other URLs in the site), so that search engines can more intelligently crawl the site. By submitting a Sitemap to a search engine, or by referencing the Sitemap in your `robots.txt` file, you can feed a list of pages into the search engine for indexing.

For more information about Sitemaps, go to:

`http://sitemaps.org/`

WebCenter Portal applications can provide a Sitemap URL that you can register with search engines to get them to index your set of pages. If the user-agent of the search engine matches one supported by ADF, the HTML is returned for the request instead of the JavaScript.

Note: Currently, the only user-agent registered with ADF is Google's googlebot.

You can create a Sitemap for your application based on a navigation model, as shown in Example 14–15.

Example 14–15 Creating a Sitemap

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          Xmlns:f=http://java.sun.com/jsf/core">
  <jsp:directive.page contentType="text/xml;charset=UTF-8" />
  <f:view>
    ${navigationContext.defaultNavigationModel.defaultSiteMap}
  </f:view>
</jsp:root>
```

15

Creating and Managing Skins

Skins define the appearance of an application. You can use skins to achieve some level of consistency across pages in your application, and to get your company's preferred look and feel.

This chapter describes how to create and manage skins. It includes the following sections:

- Section 15.1, "Overview of Skins"
- Section 15.2, "Creating a Skin in JDeveloper"
- Section 15.3, "Editing portal-skin.css"
- Section 15.4, "Enabling Runtime Administration of Skins"
- Section 15.5, "Applying Skins to WebCenter Portal: Framework Applications"
- Section 15.6, "How to Create and Manage Spaces Skins"
- Section 15.7, "Deploying a Skin as a Separate Shared Library"
- Section 15.8, "Conditionally Changing Skins for Users"
- Section 15.9, "Troubleshooting Problems with Skins"
- Section 15.10, "Files Relates to Skins"

15.1 Overview of Skins

Skins help you define the colors, fonts, images, and some dimensional details like the height and width of your application components to represent your company's preferred look and feel.

Skins are based on the Cascading Style Sheet (CSS) specification. A skin is a CSS file containing various skin selectors that define the styles of your application components. You can adjust the look and feel of any component by changing its style-related properties. Use of a skin in an application helps you to avoid specifying styles for each component individually or inserting a style sheet on each page. Every component automatically uses the styles defined in the skin. Skins help you to change an application's appearance without changing the portal pages themselves.

ADF Faces skins drive the look and feel of WebCenter Portal applications. Out-of-the-box, ADF Faces provides various built-in skins, such as `fusion`, `blafplus-rich`, `blafplus-medium`, and `simple`. By default, Framework applications use the `portal` skin, which extends the `fusionFx-v1.desktop` skin.

If the built-in skin available in your application does not meet your business requirements, you can create your own custom skins in Oracle JDeveloper. While

working with custom skins, you may need to edit the following skin-related files: `adf-config.xml`, `trinidad-config.xml`, and `trinidad-skins.xml`. For information about these files, see Section 15.10, "Files Relates to Skins."

Oracle WebCenter Portal supports runtime administration of skins to help application users continue developing a portal even after the application has been deployed. When you enable runtime administration, authorized application users can create and manage skins at runtime, without redeploying the application. You can bring runtime-created skins back into design time, edit them, and upload them into the deployed application. For information, see Section 15.4, "Enabling Runtime Administration of Skins."

In JDeveloper, you can create new skins for use in WebCenter Portal's Spaces application. In JDeveloper, you can also edit skins originally created in Spaces, and upload the edited skins back into Spaces. For information, see Section 15.6, "How to Create and Manage Spaces Skins."

15.2 Creating a Skin in JDeveloper

By default, Framework applications use the `portal` skin, which is defined in the `portal-skin.css` file. You can easily edit this file to suit your requirements. However, if you want to use a custom skin in your application, you must create a CSS file and define the required skin selectors.

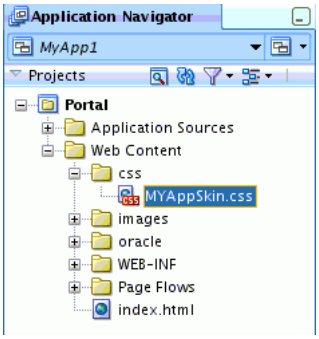
This section includes the following subsections:

- Section 15.2.1, "What You Should Know About Skin Creation"
- Section 15.2.2, "How to Create a CSS File"
- Section 15.2.3, "How to Define Skin Selectors"

15.2.1 What You Should Know About Skin Creation

When you create a CSS file, by default, JDeveloper places it under the *`APPLICATION_ROOT`*`/Portal/public_html/css` folder on the file system. In Application Navigator, the skin is displayed in the folder hierarchy as shown in Figure 15–1.

Figure 15–1 A Skin Created at the Default Location in JDeveloper

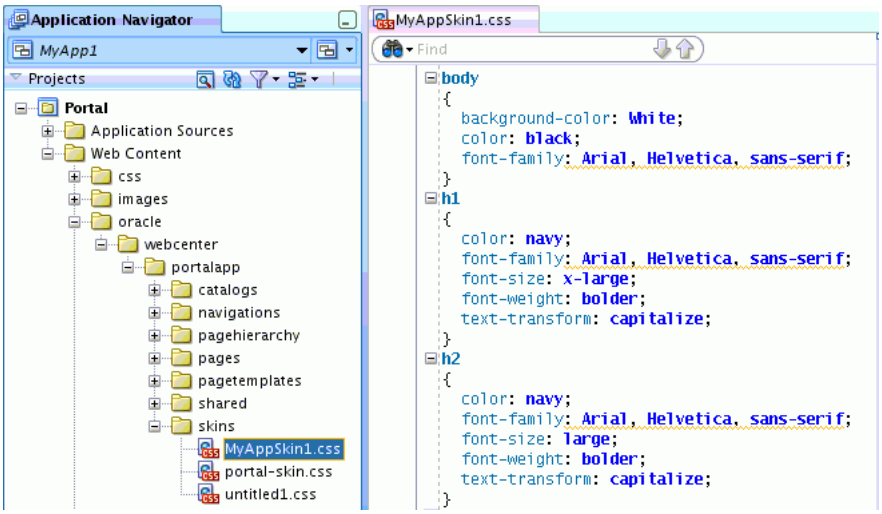


However, skins created at this default location cannot be managed at runtime. To enable authorized application users to be able to manage a skin at runtime, you must expose it as a portal resource in the Resource Manager. For a skin to be exposed as a portal resource, you must create it at the following path:

`APPLICATION_ROOT``/Portal/public_html/oracle/webcenter/portalapp/skins`

In Application Navigator, the skin is listed in the folder hierarchy as shown in Figure 15–2.

Figure 15–2 A Skin Created as a Portal Resource



After creating a CSS file, you must define the required ADF Faces skin selectors for the required components in your application. For example, you can use the `.AFDefaultFontFamily:alias` selector to specify the font family for your application as follows:

```
.AFDefaultFontFamily:alias {
font-family: Tahoma, Verdana, Helvetica, sans-serif;
}
```

The images related to a skin exposed as a portal resource must be stored at the following location:

APPLICATION_ROOT`/Portal/public_html/oracle/webcenter/portalapp/shared`

To avoid any conflicts or images getting overwritten, as a best practice you can store images in a subdirectory under the `shared` directory. To add image reference, you can use the format shown in the following example:

```
.AFBrandingBarLogo {
  background-image:
url('/oracle/webcenter/portalapp/shared/skins/images/image1.gif');
```

15.2.2 How to Create a CSS File

To create a CSS file:

1. Open your application in Oracle JDeveloper.
2. Make sure that CSS Level 3 and ADF Faces are selected.
   a. From the menu bar, choose **Tools** > **Preferences** > **CSS Editor**.
   b. From the **Support CSS Level** list, select **CSS Level 3**.
   c. For **Supported Components,** select **ADF Faces Extension**.
3. Select the **Portal** project, and from the **File** menu, choose **New**.

4. In the New Gallery dialog under **Categories**, expand **Web Tier**, then select **HTML**.
5. Select **CSS File**, and click **OK**.
6. In the Create Cascading Style Sheet dialog, in the **File Name** field, enter a name for your skin.
7. In the **Directory** field, enter the path where you want to create the CSS file. You can use the **Browse** button to specify the path.

   Note: In a Framework application, to be able to expose a skin in the runtime Resource Manager, you must create the skin under the `skins` root directory or in its subdirectory at the following path:

   APPLICATION_ROOT`/Portal/public_html/oracle/webcenter/portalapp/skins`

8. Click **OK**.

   The skin's CSS file opens in the Source View.

   You can now define the required skin selectors for the skin. For information, see Section 15.2.3, "How to Define Skin Selectors."

15.2.3 How to Define Skin Selectors

Use the ADF Faces skin selectors to define the look and feel of the different components in your application.

For information about:

- ADF Faces skin selectors in general, refer to the "Skin Style Selectors" section in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*. Also refer to Oracle JDeveloper's online help for information about the selectors that you can use in a skin. These are documented in the "Skin Selectors for Fusion's ADF Faces Components" and "Skin Selectors for Fusion's Data Visualization Tools Components" topics in JDeveloper's online help.
- Defining ADF Faces component style selectors, see sections "Defining Skin Style Properties" and "Changing the Style Properties of a Component" in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

15.3 Editing portal-skin.css

This section includes the following subsections:

- Section 15.3.1, "What You Should Know About portal-skin.css"
- Section 15.3.2, "How to Edit portal-skin.css"

15.3.1 What You Should Know About portal-skin.css

By default, a Framework application is configured to use the `portal-skin.css` skin file. This skin file provides a skeleton structure of various useful skin selectors, and contains commented-out entries that describe the purpose of each skin selector (Figure 15–3).

Figure 15–3 The portal-skin.css File



You can easily modify `portal-skin.css` to customize the default skin settings as per your requirements. You can add any new skin selectors to the file. You can also override the default settings of a skin selector by specifying the required values. For example, to set a font size of 16px for your application content, search for the `.AFDefaultFont:alias` skin selector and add the code marked in bold in the following example:

```
.AFDefaultFont:alias
{
font-size:16px;
}
```

15.3.2 How to Edit portal-skin.css

To edit `portal-skin.css`:

1. Open your application in JDeveloper.
2. In Application Navigator, navigate to the following folder path: **Portal** > **Web Content** > **oracle** > **webcenter** > **portalapp** > **skins**.
3. Right-click `portal-skin.css` and choose **Open**.
4. In the visual editor, search for the skin selector you want to edit, and add the required settings.

   For information about skin selectors, see Section 15.2.3, "How to Define Skin Selectors."
5. Save `portal-skin.css`.

15.4 Enabling Runtime Administration of Skins

This section includes the following subsections:

- Section 15.4.1, "What You Should Know About Enabling Runtime Administration of Skins"
- Section 15.4.2, "How to Enable Runtime Administration of a Skin"
- Section 15.4.3, "How to Enable Further Development of Deployed Skins"
- Section 15.4.4, "Creating a Skin in JDeveloper and Making it Available at Runtime"
- Section 15.4.5, "How to Create and Manage Skins at Runtime"

15.4.1 What You Should Know About Enabling Runtime Administration of Skins

The Resource Manager in a Framework application enables users with administrative privileges to create and modify portal resources, including skins, at runtime. By exposing skins in the Resource Manager, you allow administrators to manage these skins at runtime. If you want to create and expose more skins after deploying the application, you can export them from JDeveloper to the runtime application. At runtime, the Resource Manager provides an option to upload the skins that you exported.

Note: If you create a skin at runtime and do not import it back into the design time environment, your new skin may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To enable runtime administration of a skin, you must add it as a portal resource to the Resource Manager. For a skin to be added as a portal resource, it must be located in the *APPLICATION_ROOT*`/Portal/public_html/oracle/webcenter/portalapp/skins` directory.

While exposing a skin as a portal resource, you must set certain attributes (Figure 15–4). Table 15–1 describes the attributes that are specific to skins. For information about generic attributes, refer to Section 17.2.1, "How to Add a Resource to the Resource Manager."

Note: The Content Directory field must point to *APPLICATION_ROOT*`/Portal/public_html/oracle/webcenter/portalapp/shared` directory or a subdirectory underneath. This is the path where the dependent objects of a resource, such as images, JavaScripts, style sheets, and HTML files, must be stored. Only the objects located at this path are included in the export archive of a skin and uploaded at runtime.

Figure 15–4 Creating a Skin as a Portal Resource

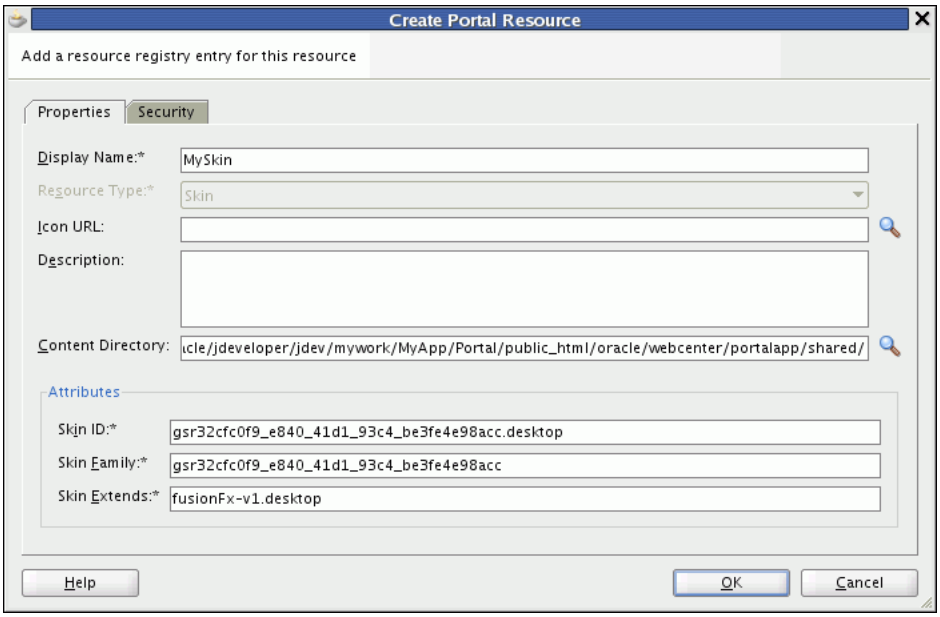


Table 15–1 Skin-Specific Attributes

Property	Description
`Skin ID`	Specifies the unique identifier of a skin. Typically, it is a combination of the skin family and the render kit, for example `BrightBlue.desktop`.
`Skin Family`	Specifies the family to which a skin belongs. It is an identifier that can be used by a number of skins with different renderkits. For example, you could have a family named `BrightBlue`. It can be used with renderkit `desktop` or `mobile`. Skin Family is really the value that `trinidad-config.xml` uses to find the matching skin in its specific attributes. You must ensure that the `Skin Family` attribute is unique. For Framework applications, the default skin is `portal-skin.css` file, with the skin family as `portal`. Ensure that you do not use `portal` as the skin family ID for any other skin.
`Skin Extends`	Specifies the ID of the skin being extended. If creating a new skin, `fusionFx-v1.desktop` is the most common skin to extend. Table 15–2 lists the skin IDs of the built-in skins available in Spaces.

Note: Skins are identified internally by skin IDs. While adding an existing custom skin as a portal resource in JDeveloper, you must not change the values of the `Skin ID`, `Skin Family`, and `Skin Extends`. Otherwise, when you upload the skin into your application, it will be included as a new skin, and the existing skin will not be replaced.

Extending Spaces Skins

You can bring your Spaces skins into JDeveloper for further development (Section 15.6, "How to Create and Manage Spaces Skins"). You can also create new skins in JDeveloper for use in the Spaces application. You create a new skin by extending an existing skin. In JDeveloper, you can extend a built-in ADF Faces skin like `blafplus-rich.desktop`, `fusion.desktop`, or `fusionFx-v1.desktop`. You can also choose to extend one of the built-in skins available in Spaces. Table 15–2 lists the skin IDs of the built-in skins available in Spaces.

Table 15–2 IDs of Built-in Skins in Spaces

Built-in Skin in Spaces	Skin ID
Fusion FX	`webcenter-fusion.desktop`
Spaces FX	`webcenterfx.desktop`

15.4.2 How to Enable Runtime Administration of a Skin

Use the following options on the context menu of a skin's CSS file to register, update, or deregister a skin in the Resource Manager:

- **Create Portal Resource**: Select to expose the skin as a portal resource. This creates an entry for the skin in the application's `generic-site-resources.xml` file. At runtime, the skin is then available for selection.

  For information, see Section 17.2.1, "How to Add a Resource to the Resource Manager." While creating a skin entry, you must specify the skin-specific attributes described in Table 15–1.

- **Update Portal Resource**: Select to edit the attributes that you defined while creating skin entries. For information, see Section 17.2.3, "How to Update Portal Resource Properties."

- **Remove Portal Resource**: Select to remove the entry for the skin from the `generic-site-resources.xml` file. For information, see Section 17.2.4, "How to Remove a Resource from the Resource Manager."

15.4.3 How to Enable Further Development of Deployed Skins

When you enable runtime administration of resources, authorized users can create new skins and modify existing ones at runtime. If users create or edit skins at runtime, you may find that you need to bring them back into JDeveloper for further enhancement. This type of round-trip development involves:

- Downloading the skin from the deployed application to get the skin's EAR file
- Importing the skin's EAR file into JDeveloper
- Editing the skin in JDeveloper
- Exporting the edited skin as an EAR file in JDeveloper
- Uploading the EAR file back into the deployed application

This section describes how to import and export skins in JDeveloper. It includes the following sections:

- Section 15.4.3.1, "Importing a Skin"
- Section 15.4.3.2, "Exporting a Skin"

For generic information about downloading and uploading a resource in a deployed application, see the sections "Downloading a Resource" and "Uploading a Resource" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

15.4.3.1 Importing a Skin

When you download a skin from a deployed application, an EAR file is created containing the skin's CSS file and the `generic-site-resources.xml` file, which contains the metadata specific to the downloaded skin.

After downloading a skin, you must import the EAR file into JDeveloper. On importing, the CSS file is extracted to the following path:

APPLICATION_ROOT`/Portal/public_html/oracle/webcenter/siteresources/scopedMD/`*scope_GUID*`/`

For detailed steps to import a resource, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

After importing a skin, you can edit its CSS file as you would any other skin file at design time. For information about editing a skin, see Section 15.2.3, "How to Define Skin Selectors." The updated skin must then be exported to the deployed application.

> **Note:** If you download a skin from a deployed application and then import and modify it in JDeveloper, you must not change values of the following attributes: `Skin Family`, `Skin ID`, and `Skin Extends`. Otherwise, when you upload the skin into the deployed application, the existing skin will not be replaced.

Note: If you edit or add images (or other dependant objects like JavaScripts and HTML files) in an imported skin, the images must be placed at the following path, which is the default content directory for skins created at runtime:

```
APPLICATION_ROOT/Portal/public_
html/oracle/webcenter/siteresources/scopedMD/shared/
```

Images must always be placed directly under or in a subdirectory of the directory set as the content directory of a portal resource. You can find out the content directory path from the Update Portal Resource dialog.

To prevent any conflicts, you may want to put your images in a subdirectory under the `shared` directory. For example, you may put images in the `skins/images` subdirectory. In such a case, you must update the content directory path to point to the new location.

Once you add your images, you can reference them in the CSS file as shown in the following example:

```
.AFBrandingBarLogo {
background-image:
url('/oracle/webcenter/siteresources/scopedMD/shared/skins/images/i
mage1.gif');
}
```

For more information, see Section 15.2.1, "What You Should Know About Skin Creation."

15.4.3.2 Exporting a Skin

You may need to export a skin in the following situations:

- To copy back a skin that you imported from a deployed application and modified it.
- To expose a new skin in a deployed application.

When you export a skin in JDeveloper, an EAR file is created containing the skin's CSS file and the `generic-site-resources.xml` file. Users with administrator privileges can then upload this EAR file to a deployed application by using the Resource Manager. Once uploaded, the skin can be managed and applied at runtime.

For detailed steps to export a resource, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

15.4.4 Creating a Skin in JDeveloper and Making it Available at Runtime

Even after you have deployed your Framework application, you can create new skins at design time and make them available at runtime. This involves the following steps:

1. Creating the skin in JDeveloper, as described in Section 15.2, "Creating a Skin in JDeveloper."
2. Adding the skin as a portal resource by using the Create Portal Resource option, as described in Section 15.4.2, "How to Enable Runtime Administration of a Skin."
3. Exporting the skin as a portal resource in JDeveloper, as described in Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

4. Uploading the skin to the runtime application using the Resource Manager, as described in the section "Uploading a Resource" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

15.4.5 How to Create and Manage Skins at Runtime

With runtime administration enabled in your Framework application, authorized users can use the Resource Manager to perform the following tasks: create a new skin, edit a skin, edit the source code, copy, set security, show or hide, set attributes, delete, upload, and download a skin.

Figure 15–5 shows the Resource Manager, with the Edit menu for a skin displayed.

Figure 15–5 Managing Skins at Runtime



For information about how to access the Resource Manager, see Section 17.3.2, "How to Use the Resource Manager at Runtime." The procedures for working with skins in a deployed Framework application are same as those in the Spaces application. For generic information about creating and managing a skin at runtime, see the "Working with Skins" chapter in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

15.5 Applying Skins to WebCenter Portal: Framework Applications

You can choose to apply a different skin to your Framework application. If runtime administration is enabled, authorized users can change the skin at runtime.

This section includes the following subsections:

- Section 15.5.1, "Applying a Skin at Design Time"
- Section 15.5.2, "Applying a Skin at Runtime"

15.5.1 Applying a Skin at Design Time

At design time, to switch to a different skin that has been created as a portal resource, directly edit the `adf-config.xml` file. For information about the attributes that you can set in `adf-config.xml`, see Section 15.10.1, "adf-config.xml."

To apply a skin created as a portal resource to your Framework application at design time:

1. In JDeveloper, open `adf-config.xml` of your application.

To locate this file, in the Application Navigator, expand **Application Resources** > **Descriptors** > **ADF META-INF**.

2. Under `<portal:preferences>`, search for the following skin properties:

```
<portal:preference id="oracle.webcenter.portalapp.skin"
                   desc="Default Portal Skin" value="portal"
                   resourceType="Skin" display="true"/>
```

3. In the `value` property, specify the family name of the skin that you want to apply to your application.

   This value must be same as the value specified for the `Skin Family` attribute in the Create Portal Resource or Update Portal Resource dialog of the required skin.

4. Optionally, in the `desc` property, specify a description of the skin you want to apply.

5. Save `adf-config.xml`.

Note: If you want to apply a skin created at any other location other than *APPLICATION_ROOT*`/Portal/public_html/oracle/webcenter/portalapp/skins`, you must register the skin in the `trinidad-skins.xml` file and configure your application to use the skin. For information, see the chapter "Customizing the Appearance Using Styles and Skins" in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

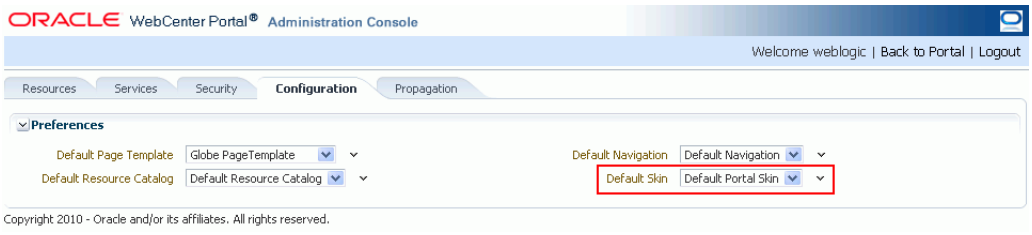
15.5.2 Applying a Skin at Runtime

If you enable runtime administration of resources, you do not necessarily need to manually register a skin for your application. You can apply the desired skin at runtime in your deployed application.

To apply a skin at runtime in a deployed Framework application:

1. Log on as the application administrator, and navigate to the WebCenter Portal Administration Console.
2. Click the **Configuration** tab.
3. From the **Default Skin** list, select the desired skin Figure 15–6.

Figure 15–6 Applying a Skin at Runtime



15.6 How to Create and Manage Spaces Skins

Out-of-the-box, WebCenter Portal's Spaces application provides certain default skins. If these default skins do not suit your requirements, you can build your own skins

within Spaces. For information, see the chapter "Working with Skins" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

However, Spaces may not provide all the controls to create a skin with all the functionality that you require. In such cases, you can create your own skin in JDeveloper and then upload it into Spaces. You can also download an existing skin from Spaces and edit it in JDeveloper, or use one of the sample skins as your starting point, and then upload it back into Spaces.

WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, including skins, to Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build custom skins and other types of resources for Spaces.

15.7 Deploying a Skin as a Separate Shared Library

When you use a custom skin for your Framework application, as a best practice you must deploy the custom skin as a separate shared library. This is especially important when you want your Framework application to consume a portlet that renders using the same skin as your application.

15.7.1 About Skin Sharing

WebCenter Portal: Framework provides various default skins. All WebCenter Portal skin artifacts are available in the shared library, `oracle.webcenter.skin`. The shared library is packaged as an EAR, and includes several skin JARs that contain the following types of files:

- `.css` file, the skin selector file
- `.png`, the image files
- `.js`, the javascript file
- `trinidad-skins.xml`, the skin definition file

By default, the `oracle.webcenter.skin` shared library is referenced by all Framework applications and Portlet Producer applications. Therefore, all skins are available through the classpath. When you use a portlet in your Framework application, the portlet is rendered using the skin defined for your application. Therefore, to achieve skin sharing, the same skin must be accessible through the shared library, to both the Framework application and the Portlet Producer application.

However, there may be a case where you modified `portal-skin.css` in your Framework application. So, the skin available in the shared library is not the same as the updated version available in the Framework application. If the skin's version in the Framework application is not identical to the skin in the shared library, skin sharing between Framework application and Portlet Producer application will fail because the Portlet producer can access only the original version of the skin packaged in the shared library, while the Framework application uses the modified skin. The Framework application renders with the latest skin changes, while the portlet within the same application renders with the original skin from the shared library. This behavior is called skin coordination.

Instead of using the default skin, you may create a new custom skin in your Framework application, either as a portal resource or define it in the `trinidad-skins.xml` with a unique skin family ID. If your application uses a portlet, the portlet producer will not be able to find the custom skin in the shared library. This causes a skin mismatch problem, and the portlet is rendered with a bare bone, simple skin rather than the custom skin used by the Framework application.

Therefore, whenever you modify a skin or create a custom skin for your Framework applications, you must deploy the new or modified skin as a shared library.

15.7.2 How to Deploy a Skin as a Separate Shared Library

To deploy a skin as a shared library:

1. Ensure that a copy of `trinidad-skins.xml` is available in the `WEB-INF` folder of your Framework application. If not, create a new one.
2. Update `trinidad-skins.xml` to point to the newly created skin CSS file.
3. Create a new `MANIFEST.MF` file in the project and provide the details for the new shared library.

   The following is a sample code in a `MANIFEST.MF` file:

   ```
   Manifest-Version: 1.0
    Ant-Version: Apache Ant 1.7.1
    Created-By: 1.6.0 (Sun Microsystems, Inc.)
    Extension-Name: oracle.webcenter.customskin
    Implementation-Title: oracle.webcenter.customskin
    Implementation-Version: 11.1.1
    Implementation-Vendor: Oracle
    Specification-Title: Custom WebCenter shared library
    Specification-Version: 11.1.1
   ```

4. Create a new deployment profile:
   a. Right-click the **Portal** project, then choose **New**.
   b. In the New Gallery dialog, choose **General** > **Deployment Profile** > **Shared Library JAR File**, and click **OK**.
   c. In the Create Deployment Profile dialog, in the **Deployment Profile Name** field, enter the profile name and click **OK**.
   d. In the Edit JAR Deployment Profile Properties dialog, ensure that **Include Manifest File** checkbox is selected.
   e. In the **Additional Manifest Files to Merge into MANIFEST.MF** section, use the **Add** button to add `trinidad-skins.xml` and the desired skin CSS file in the shared library.
   f. Click **OK**.
   g. In the Project Properties dialog, click **OK**.
5. Right-click the **Portal** project, choose **Deploy**, then select the JAR shared library deployment profile name.
6. In the Deploy dialog, select **Deploy to a Weblogic Application Server**, and click **Next**.
7. Select the WebLogic Server that hosts your portlet producer, and click **Next**.
8. Ensure that **Deploy as a Shared Library** is selected, then click **Finish**.

9. Update `/META-INF/weblogic-application.xml` for both the Framework application and the Portlet Producer application to reference the custom skin shared library.

```
<library-ref>
    <library-name>Custom_Library_Name</library-name>
    <specification-version>Library_Version</specification-version>
  </library-ref>
```

10. Redeploy both the Framework application and the Portlet Producer application.

    The custom skin can now be shared between these applications.

> **Note:** The custom skin shared library must be deployed, and must appear as "Active" on the WebLogic Server console before true skin sharing can be achieved. If there is any issue with the shared library JAR (like incorrect manifest or incorrect version), the deployed shared library will not show up in the WebLogic Server console as "Active".

> **Note:** If you have multiple skin files, you can put all skin artifacts into one shared library JAR, and merge the two `trinidad-skins.xml` files into a single file. You can keep using this shared library for all your custom skins development.

15.8 Conditionally Changing Skins for Users

You can assign different skins per user, page, application, and so on, without impacting the actual application logic. To conditionally set a skin, you use the `<skin-family>` entry in the `trinidad-config.xml` file.

You can use EL expressions that can be evaluated dynamically to determine the skin to display. For example, if you want to use the German skin when the user's browser is set to the German locale, and to use the English skin otherwise, use the following `<skin-family>` entry in the `trinidad-config.xml` file:

```
<skin-family>#{facesContext.viewRoot.locale.language=='de' ?
'german' : 'english'}</skin-family>
```

For more information, see the "How to Configure a Component for Changing Skins Dynamically" section in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

> **Note:** The default value of `<skin-family>` is `#{preferenceBean.defaultTrinidadSkin}`. If you change the default value, your application users cannot set skins at runtime in the Resource Manager.

You can also use the `SkinSetting` API to set the skin for a user conditionally. Refer to the API for information about the calls you can use. The API is available here:

http://download.oracle.com/docs/cd/E15523_01/apirefs.1111/e15995/oracle/webcenter/generalsettings/model/SkinSetting.html

15.9 Troubleshooting Problems with Skins

This section provides information to assist you in troubleshooting problems you may encounter while using skins.

Problem

Your application does not reflect skin changes made at design time.

Solution

During development, after you make changes to a skin, you can see your CSS changes without restarting the server. The `CHECK_FILE_MODIFICATION` context parameter requires the server to check the timestamp on a skin and reload it if the skin has changed. This setting is automatically set to `true` if iterative development is enabled for your application. If iterative development is not enabled, ensure that `CHECK_FILE_MODIFICATION` is set to `true` in `WEB-INF/web.xml`, as shown in the following example:

```
<context-param>
<param-name>org.apache.myfaces.trinidad.CHECK_FILE_MODIFICATION</param-name>
<param-value>true</param-value>
</context-param>
```

Problem

Skin selectors appear encoded and compressed when you open a skin's CSS file in a browser.

Solution

By default, class names in a CSS file are compressed to reduce the overall size of the file. So, the skin selectors you see, for example in the Firebug extension of Firefox, are encoded values bearing little relationship to the original names. For example, a CSS file may contain an entry that looks like:

```
 .x123 {color: #534741}
```

It is easier to examine a CSS file when the compression is turned off. You can disable compression by setting the value of `DISABLE_CONTENT_COMPRESSION` context parameter to `true` in `WEB-INF/web.xml`, as shown in the following code:

```
<context-param>
<param-name>org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION</param-name>
<param-value>true</param-value>
</context-param>
```

With the compression turned off, the entry may look like:

```
.af_panelFormLayout_label-cell
{
color: #534741
}
```

For better performance, it is recommended that you turn on CSS compression in a production environment. If you turn off CSS compression, it may lead to skin mismatching, as described in the next problem scenario.

Problem

When you run a page containing a portlet, the portlet is rendered without any styles or formatting.

Solution

When your application consumes a portlet based on an ADF application, the portlet producer tries to match the skin of your application. If the skins match, the producer uses your application's skin.

Skin compression is not taken into account during skin matching. If the skin compression settings differ but the skins match, the producer returns IDs like ".x123" and expects to find these values in your application; whereas your application may return values like ".af_panelFormLayout_label-cell". Therefore, no styles are found and the portlet rendering fails.

Your application and portlet producers must use the same skin compression settings. If your application uses an uncompressed skin, your portlet producer also must use an uncompressed skin.

Problem

Style sheet files (CSS) within the MDS folder structure (`oracle/webcenter/portalapp`) do not take effect as specified in `trinidad-skins.xml`.

Solution

For style sheet files (CSS) within the MDS folder structure, prefix "`mds:`" to the path in `trinidad-skins.xml`, as shown in the following example:

```
<skin>
   <id>myskin2.desktop</id>
   <family>myskin2</family>
   <render-kit-id>org.apache.myfaces.trinidad.desktop</render-kit-id>
 <style-sheet-name>mds:/oracle/webcenter/portalapp/css/skin.css</style-sheet-name>
   <extends>fusion.desktop</extends>
</skin>
```

15.10 Files Relates to Skins

This section includes the following subsections that describe the files that are created or modified when you create or modify skins:

- Section 15.10.1, "adf-config.xml"
- Section 15.10.2, "trinidad-config.xml"
- Section 15.10.3, "trinidad-skins.xml"

15.10.1 adf-config.xml

You can leverage the Framework application's preference mechanism, and there is built-in support for defining a preference for skins to be used at runtime, in `adf-config.xml`. This file is located in the `META-INF` directory. Example 15–1 shows the default skin preferences available in `adf-config.xml`.

Example 15–1 Skin Preferences in adf-config.xml

```
<portal:preference id="oracle.webcenter.portalapp.skin"
```

```
desc="Default Portal Skin" value="portal"
resourceType="Skin" display="true"/>
```

In `adf-config.xml`, you can define the following properties for your skin:

- `preference id` - the hard-coded value used in the Portal Preferences API as a key to lookup the preference value of the skin to be used in an application.
- `desc` - description of the skin. At runtime, this value shows up as the display name of the skin.
- `value` - the family name of the skin used in the application. This value must be same as the value specified for the `Skin Family` attribute in the Create Portal Resource or Update Portal Resource dialog of the required skin. If you want your application to use a different skin, specify the required skin family name here.
- `resourceType` - the type of the resource; in this case, a skin.
- `display` - specifies whether the skin is listed in the skin picker at runtime.

15.10.2 trinidad-config.xml

In JDeveloper, when you create a Framework application, a starter `trinidad-config.xml` file is automatically created in the `/WEB-INF` directory. Example 15–2 shows a starter `trinidad-config.xml` file.

Example 15–2 Starter trinidad-config.xml File Created by JDeveloper

```
<?xml version="1.0" encoding="US-ASCII"?>
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">
  <skin-family>#{preferenceBean.defaultTrinidadSkin}</skin-family>
</trinidad-config>
```

In the `trinidad-config.xml`, you can set `<skin-family>` that determines which skin to use, and if necessary, under what conditions. The `trinidad-config.xml` file has a simple XML structure that enables you to define element properties using the JSF Expression Language (EL) or static values.

Note: The default value of `<skin-family>` is `#{preferenceBean.defaultTrinidadSkin}`. If you change the default value, your application users cannot set skins at runtime in the Resource Manager.

15.10.3 trinidad-skins.xml

The `trinidad-skins.xml` file is required for any skin that is provided as a JAR. The file is usually bundled with each skin JAR to provide a lookup between skin ID and the actual path of the skin. However, for a skin created as a portal resource in a Framework application, a separate runtime API is used to lookup the location of a skin. Therefore, this file is not needed for such skins.

For more information about `trinidad-skins.xml`, see the chapter "Customizing the Appearance Using Styles and Skins" in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

16

Creating and Managing Resource Catalogs

Resource Catalogs define and organize the resources available for inclusion in a page, page template, or task flow. You can create them in JDeveloper so that they are deployed with the application. Resource Catalogs can also be created and edited in the runtime Resource Manager. This chapter describes the default Resource Catalog and explains how to customize this catalog and create new catalogs in JDeveloper.

For information about managing Resource Catalogs at runtime, see "Working with Resource Catalogs" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

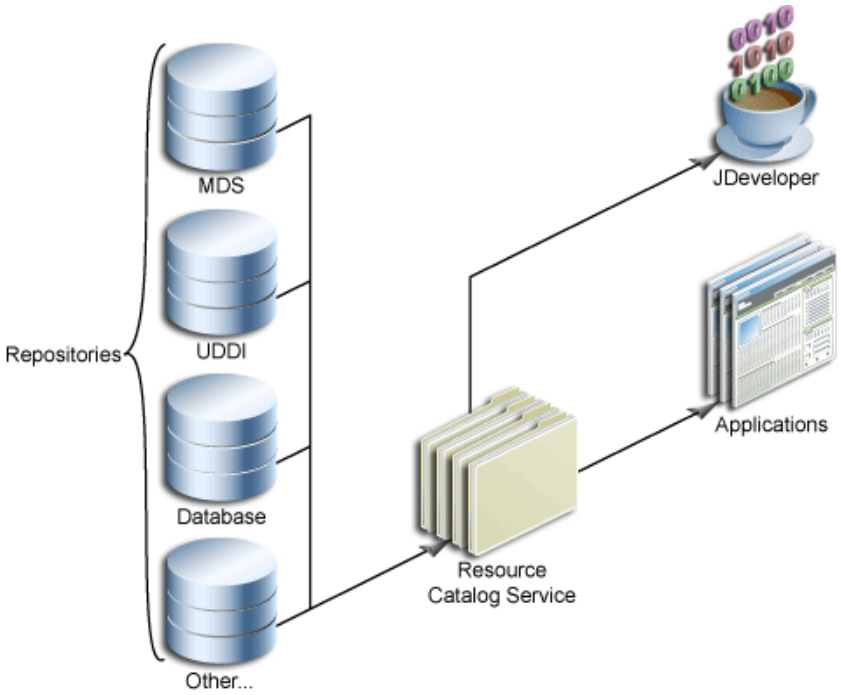
This chapter contains the following sections:

- Section 16.1, "Overview of Resource Catalog"
- Section 16.2, "Creating a Custom Resource Catalog"
- Section 16.3, "Managing a Resource Catalog and Enabling Runtime Administration"
- Section 16.4, "Filtering Items in the Resource Catalog"
- Section 16.5, "Configuring Multiple Resource Catalogs"

16.1 Overview of Resource Catalog

The Resource Catalog, also known as *Oracle Business Dictionary*, provides a consolidated view of the contents of one or more otherwise unrelated repositories in a unified search and browse user interface. Resources originate in their source repository and are then exposed through the Resource Catalog as shown in Figure 16–1.

Figure 16–1 Resource Catalog Overview



When using JDeveloper, you see two types of Resource Catalogs—design time Resource Catalogs and runtime Resource Catalogs.

Design time Resource Catalogs are available in the Resource Palette and are like favorites lists for developers. The WebCenter Portal - Services Catalog is a design time catalog that is available out-of-the-box when you install the WebCenter Portal extension bundle. You can add task flows and data controls from this catalog to a Framework application. In addition to the WebCenter Portal - Services Catalog, you can define connections and create catalogs to organize resources exposed by those connections. Such resources can be re-used in any application that you are developing.

See Also:

- Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper"
- Oracle JDeveloper online Help for more information about the Resource Palette and catalogs.

  To locate this information in Oracle JDeveloper, from the **Help** menu, select **Table of Contents**. In Help Center, expand **JDeveloper Basics** and select **Working with the Resource Palette**.

Runtime Resource Catalogs are available for populating pages or page templates and task flows in deployed Spaces and Framework applications. In Spaces applications, the Resource Catalogs determine the resources available when populating pages and task flows in Composer. In addition to this, the Resource Catalogs in Framework applications also determine the resources available when creating page templates and other Resource Catalogs in the Resource Manager.

See Also:

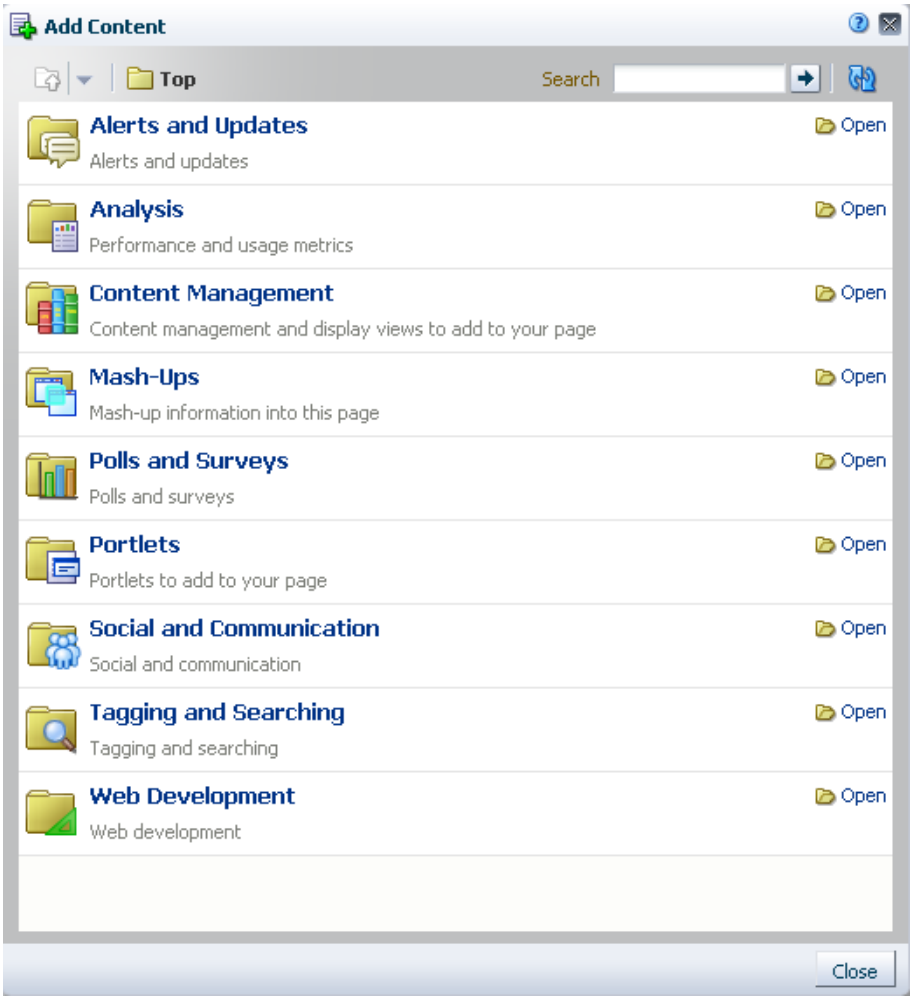
- Section 20.1, "Designing Editable Pages Using Composer Components"
- Section 12.2, "Working with Page Templates"

Default Resource Catalog Configuration

When you add a `Page Customizable` component to the page, a default Resource Catalog with the catalog definition file, `default-catalog.xml`, is configured in the application. Depending on the type of application you created, the catalog definition file is created in one of the following folders:

- In a Framework application, the `default-catalog.xml` file is located in the *`APPLICATION_ROOT`*`/Portal/public_html/oracle/webcenter/portalapp/catalogs` directory. The Resource Catalog in this case contains the folders shown in Figure 16–2.

Figure 16–2 Default Resource Catalog



- In a non-portal WebCenter application, the `default-catalog.xml` file is located in the *`APPLICATION_ROOT`*`/Portal/src/portal` directory. The Resource Catalog in this case contains the **ADF Faces Components** folder, which contains ADF Faces components that a user can add to the page.

If you have registered a portlet producer with your application, then that producer's portlets are displayed in a Portlets folder in the Resource Catalog. If you have not registered any producer, the Portlets folder is hidden.

You can create custom catalogs or modify the default catalog and include components relevant to your business. For more information, see Section 16.2, "Creating a Custom Resource Catalog" and Section 16.2.3, "How to Manage Content in the Resource Catalog."

16.2 Creating a Custom Resource Catalog

Most businesses may require that users and groups have access to only those resources that they can add to a page. To address this need, you can configure multiple Resource Catalogs in your application. To configure multiple Resource Catalogs, you must first create custom catalogs with all the required resources.

This section describes how to create a custom Resource Catalog. It contains the following subsections:

- Section 16.2.1, "How to Create a Custom Resource Catalog"
- Section 16.2.2, "How to Define Connections to Resources"
- Section 16.2.3, "How to Manage Content in the Resource Catalog"
- Section 16.2.4, "How to Expose Data Controls Created at Design Time"
- Section 16.2.5, "How to Expose Data Controls Created at Runtime"
- Section 16.2.6, "How to Control Visibility of Portlets in the Resource Catalog"
- Section 16.2.7, "How to Configure the Default Resource Catalog for an Application or Page"
- Section 16.2.8, "What You May Need to Know When Defining a Resource Catalog"
- Section 16.2.9, "How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows"

16.2.1 How to Create a Custom Resource Catalog

The New Gallery dialog provides an option to create a Resource Catalog. Using this option, you can create a blank catalog and populate it with resources of your choice.

To create a catalog definition:

1. Open your application in Oracle JDeveloper.
2. Select the Portal project, and from **File** menu, choose **New**.
3. In the New Gallery dialog, expand **Web Tier**, select **Portal**, then **Application Resource Catalog**, and click **OK**.
4. In the File Name field in the Create Application Resource Catalog dialog, provide a name for the catalog, for example, `users-catalog.xml`.
5. In the Directory field, specify the location for the new catalog definition XML file.

   Catalog definition files must be saved in the Resource Catalog root directory or a subdirectory of the root. The default root directory in a Framework application is *`APPLICATION_ROOT`*`/Portal/public_html/oracle/webcenter/portalapp/catalogs`, and in a non-Framework application is *`APPLICATION_ROOT`*`/Portal/src/portal`.

> **Note:** In a Framework application, to expose the new catalog in the runtime Resource Manager, you must create the catalog in the *APPLICATION_ROOT*/Portal/public_html/oracle/webcenter/portalapp directory.

For example, if the Resource Catalog root is:

```
C:\JDeveloper\mywork\RCSampleApp\Portal\public_
html\oracle\webcenter\portalapp\catalogs
```

You could create your catalog here or create a subdirectory for it. For example:

```
C:\JDeveloper\mywork\RCSampleApp\Portal\public_
html\oracle\webcenter\portalapp\catalogs\mycatalogs
```

6. Select the Create as Portal Resource option to expose the catalog in the Resource Manager.

> **Note:** This option is displayed in a Framework application if you specified the directory to be *APPLICATION_ROOT*/Portal/public_html/oracle/webcenter/portalapp or a subdirectory of portalapp.

7. Click **OK**.

   The new catalog definition file opens in the Overview Editor.

8. In the right half of the Overview Editor, define catalog properties as follows:
   - In the Catalog Filter field, specify the name of the filter you want to apply to the catalog. This field is optional. For more information, see Section 16.4, "Filtering Items in the Resource Catalog."
   - In the Visible field, specify whether the Resource Catalog must be displayed. This attribute takes an EL value. The default value is `#{true}`, which means that the catalog is displayed to all users at all times.
   - Use the Catalog Attributes section to define the following attributes on the catalog.
     - **Title**: A display name for the catalog.
     - **Description**: A short description of the catalog. This text appears below the catalog name in the Resource Manager.
     - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
     - **IconURI**: Specify the URI for an icon to display alongside the catalog name in the Resource Manager.
     - **Tool Tip**: Use to specify the tool tip to be displayed when you move the mouse over this catalog name.

> **Note:** Catalog attributes are defined within the `<schema>` element in the catalog definition XML file. In addition to the out-of-the-box attributes, you can expose custom attributes on the catalog by defining them within the `<schema>` element.

9. Save the catalog definition file.

You can now populate your catalog with resources. See Section 16.2.3, "How to Manage Content in the Resource Catalog" for details about adding different types of resources.

> **Tip:** If you do not want to add resources from scratch, as a starting point, you could copy the contents of `default-catalog.xml` into your new catalog definition.

16.2.2 How to Define Connections to Resources

Many resources, such as WebCenter Portal services and portlets, are accessed through connections defined in your application's `connections.xml` file. Therefore, you must create connections before adding such resources to the Resource Catalog.

You can create connections to resources in different ways. This section describes how to create a connection from the Resource Palette. For information about additional ways to create connections, see Section 7.2, "Preparing Your Framework Application to Consume Services" and Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."

To create a connection from the Resource Palette:

1. From the Resource Palette, click the **New** icon in the upper left corner.
2. Choose **New Connection** and then the type of connection you want to create.

   The wizard for your connection type appears.
3. Step through the wizard filling in the necessary information about the connection.

   When you click **Finish**, the connection should appear in the Resource Palette under **Connections**.
4. Right click your connection in the Resource Palette and choose **Add to Application** from the context menu.

   Alternatively, you can drag and drop the connection from the Resource Palette to the Application Resources panel of the Application Navigator.

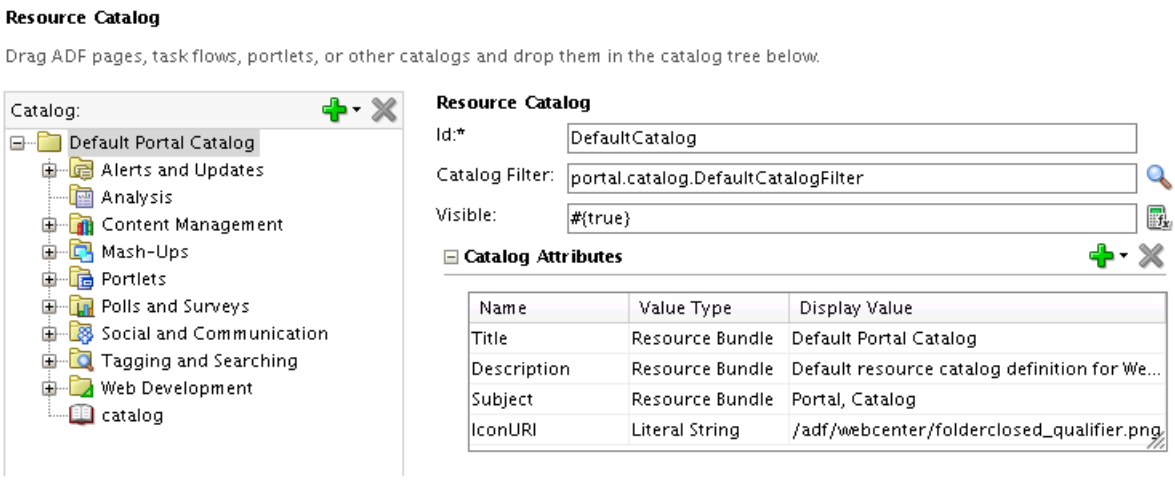
16.2.3 How to Manage Content in the Resource Catalog

You can add or remove catalog resources as required by your business.

> **Tip:** If you are creating a Resource Catalog from scratch, you can refer to the default catalog definition file to get an idea of how resources are included in the catalog.

Typically, catalog definition files are available in the *`APPLICATION_ROOT`*`/Portal/public_html/oracle/webcenter/portalapp/catalogs` directory. The Design view of a catalog definition file, shown in Figure 16–3, provides options to add, modify, and delete catalog content.

Figure 16–3 Catalog Definition File Design View



You can add resources to a catalog in two ways:

- Use the options on the Add menu in the Resource Catalog Overview Editor.
- Drag and drop resources from different areas in the IDE onto the catalog definition file.

The left half of the XML editor contains the Catalog section, which displays the hierarchical structure of components in the catalog. The right half of the page provides fields to define attributes and parameters on a selected resource.

Note: If you are using a standalone version of Composer (that is, Composer without WebCenter Portal: Framework), you can add a resource to the catalog by directly adding a `<resource>` element to the source of the catalog definition file. For more information, see Section C.3.12, "resource."

See Also: For details about the catalog definition file and its attributes:

- Section C.2, "XML Schema"
- Section C.3, "Catalog Definition Attributes"

This section explains how to manage catalog resources. It contains the following sections:

- Section 16.2.3.1, "Adding a Folder"
- Section 16.2.3.2, "Adding a Link"
- Section 16.2.3.3, "Adding a Resource Catalog"
- Section 16.2.3.4, "Adding a Component"
- Section 16.2.3.5, "Adding a Custom Folder"
- Section 16.2.3.6, "Adding a Custom Content Provider"
- Section 16.2.3.7, "Dragging and Dropping a Resource into the Catalog"
- Section 16.2.3.8, "Editing a Resource"

- Section 16.2.3.9, "Deleting a Resource"
- Section 16.2.3.10, "Rearranging Resources in the Catalog"

16.2.3.1 Adding a Folder

You can create folders to group similar resources, thereby organizing the catalog better.

To add a folder:

1. Open the catalog definition file in JDeveloper.

   This file is typically available in the *APPLICATION_ROOT*`/Portal/public_html/oracle/webcenter/portalapp/catalogs` directory.

2. From the **Add** menu in the Catalog section, select **Folder**.

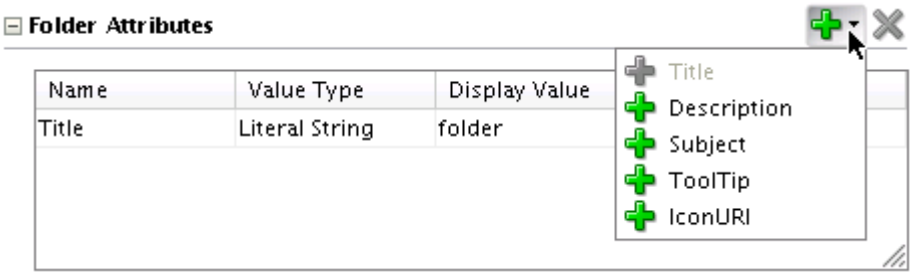
   A `folder` element is added to the catalog. The right side of the page displays fields to define the new folder's attributes and parameters.

3. The `Id` field is populated with the value `folder` by default. You can specify a different value, but the ID must be unique within the catalog definition file.

4. In the `Visible` field, you can specify an EL value to conditionally display this folder to specific users or groups. This attribute takes an EL value. The default value is `#{true}`, which means that the resource is displayed in the catalog at all times.

5. Optionally, in the Folder Attributes section, you can define the following attributes by choosing them from the **Add** menu, shown in Figure 16–4:

Figure 16–4 Folder Attributes



   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

   Selecting an option on the Add menu creates a new row in the Folder Attributes table. You can use this row to specify the name, type, and value for the new attribute.

16.2.3.2 Adding a Link

You can add the following types of links to a catalog:

- **Task Flow**: Link to a WebCenter Portal service task flow (in the Resource Palette) or a custom task flow created in Oracle JDeveloper (in the Application Navigator).
- **Portlet**: Link to any registered portlet producer. See Chapter 64, "Consuming Portlets" for detailed information about registering portlet producers and adding portlets to the page.
- **Content**: Link to a file or directory from an existing Content Repository connection.
- **Other**: Link to a custom component that you create by providing the XML code for that component.

To add a link:

1. From the **Add** menu in the Catalog section, select **Link**.

   A `url` element is added to the catalog. The right side of the page displays fields to define the new link's attributes and parameters.

2. The `Id` field is populated with the value `url` by default. You can specify a different value, but the ID must be unique within the catalog definition file.

3. From the Type list, shown in Figure 16–5, select the type of resource that you want to create. You can choose from `Taskflow`, `Portlet`, `Content`, and `Other`.

Figure 16–5 Types of Links



Notes:

- You can also add task flows and portlets to the catalog by dragging and dropping them from the Resource Palette or Application Resources pane onto the Catalog section in the Design view of the catalog definition file. The associated attributes are populated automatically when the resource is added to the catalog. See Section 16.2.3.7, "Dragging and Dropping a Resource into the Catalog."
- If you added an ADF task flow to the catalog and want to export the catalog to an already deployed application, then you must deploy the task flow as a shared library to the deployed application. For more information, see Section 13.3, "Adding Custom Task Flows to Your Portal."

4. If you are adding a custom resource, then specify the factory class of the component. For all other resource types, the `Factory Class` value is populated as soon as you select the type.

   The factory classes for the available resource types are as follows:

- Task flow: `oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourceFactory`
- Portlet: `oracle.webcenter.portalframework.sitestructure.rc.PortletResourceFactory`
- Content: `oracle.webcenter.content.model.rc.ContentUrlResourceFactory`

For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

5. In the URL field, specify the URL to access the resource. Table 16–1 provides the URL format to be used for each resource type.

Table 16–1 URL Format for the Different Resource Types

Resource Type	Location in the Application	URL Format
Task Flow from an ADF Library.	Resource Palette	**`taskflow://`***`Path_to_Task_Flow/Task_Flow_Definition_File_Name#Task_Flow_ID`*
Task Flow Within the Application.	Application Navigator	**`taskflow://`***`Path_to_Task_Flow/Task_Flow_Definition_File_Name#Task_Flow_ID`*
Portlet	Portlet producer connection in the Application Resources pane or in the Resource Palette	**`portlet://`***`Producer_ID/Portlet_ID`*
Content	Content Repository connection in the Application Resources pane	**`content://`***`Content_Connection_ID/Document_ID`*

Click the **Browse** icon next to the URL field to open the Choose Resource dialog, which enables you to add a resource of the selected type from the application or the Resource Palette.

Note: If you entered the URL for a portlet or content type resource manually, and if the connection for that resource is available only in the Resource Palette, then ensure that you first add the connection to the application.

---

6. In the `Visible` field, you can specify a true or false value, or an EL value to conditionally display this link to specific users or groups. The default value is `#{true}`, which means that the resource is displayed in the catalog at all times.
7. Optionally, in the URL Attributes section, you can define the following attributes by choosing them from the **Add** menu:
   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

Selecting an option on the Add menu creates a new row in the Folder Attributes table. You can use this row to specify the name, type, and value for the new attribute.

8. Use the URL Parameters section to define parameters on the resource. You can bind these parameters to other artifacts in the application. Click the **Add** icon to create a new row in the Parameters table.

16.2.3.3 Adding a Resource Catalog

You can include another Resource Catalog inside the selected catalog. When you add another catalog, only a pointer to this catalog is included in the catalog definition file. The actual catalog contents are inserted only at runtime.

To add a Resource Catalog:

1. From the **Add** menu in the Catalog section, select **Catalog Reference**.

   A new node titled `catalog` is added to the catalog. The right side of the page displays fields to define the new folder's attributes and parameters.

2. The `Id` field is populated with the value `catalog` by default. You can specify a different value, but the ID must be unique within the catalog definition file.

3. In the Path field, specify the MDS path to the catalog definition file, for example, `\oracle\webcenter\portalapp\catalogs\custom-catalog.xml`.

   You can use the Browse icon to browse application folders and select the catalog definition file.

4. In the `Visible` field, you can specify an EL value to conditionally display this catalog to specific users or groups. This attribute takes an EL value. The default value is `#{true}`, which means that this resource is displayed in the catalog at all times.

5. Optionally, in the Catalog Attributes section, you can define the following attributes by choosing them from the **Add** menu:

   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

   Selecting an option on the Add menu creates a new row in the Catalog Attributes table. You can use this row to specify the name, type, and value for the new attribute.

16.2.3.4 Adding a Component

You can add a custom component to the catalog. You can add a variety of components, such as simple ADF Faces components, compound objects containing two or more components, and JSF Verbatim tags that let you add arbitrary HTML content inside them. An example for arbitrary HTML that you can add to your catalog is a YouTube video.

To add a custom component:

1. From the **Add** menu in the Catalog section, select **Component**.

A `component` element is added to the catalog. The right side of the page displays fields to define the component's attributes and parameters.

2. The `Id` field is populated with the value `component` by default. You can specify a different value, but the ID must be unique within the catalog definition file.
3. In the Component Factory field, specify the name of a Java class that implements the `oracle.adf.rc.component.ComponentFactory` interface and creates an instance of the component based on a set of parameters, for example, `oracle.adf.rc.component.XmlComponentFactory`.

   As you type the name, the field auto-completes with valid factory classes from the classpath. You can also use the Browse icon to select a class.

   For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."
4. In the `Visible` field, you can specify an EL value to conditionally display this component to specific users or groups. This attribute takes an EL value. The default value is `#{true}`, which means that this resource is displayed in the catalog at all times.
5. Optionally, in the Component Attributes section, you can define the following attributes by choosing them from the **Add** menu:
   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

   Selecting an option on the Add menu creates a new row in the Catalog Attributes table. You can use this row to specify the name, type, and value for the new attribute.
6. Use the Component Parameters section to add parameters that were defined when implementing the factory class.

   For example, if you are adding an XML component, click the **Add** icon in the Parameters section and select **xml**. In the new row that is displayed, specify the XML code for the component in the value field.

   Note: The xml option is displayed only if you selected `oracle.adf.rc.component.XmlComponentFactory` as the component factory.

   To ensure that the component works properly, you must specify the custom component ID to be `#`, and provide the namespace, as shown in the following example:

```
<cust:panelCustomizable id="#"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/>
```

   The `#` value ensures that a unique ID is generated dynamically for the component.

16.2.3.5 Adding a Custom Folder

You can add a folder using a Resource Catalog adapter so that the folder is populated dynamically. A custom folder is not added as a folder with resources to the catalog; it contains only a reference to the factory class, which displays the resources dynamically.

To add a custom folder:

1. From the **Add** menu in the Catalog section, select **Custom Folder**.

   A `customFolder` element is added to the catalog. The right side of the page displays fields to define the folder's attributes and parameters.

2. The `Id` field is populated with the value `customFolder` by default. You can specify a different value, but the ID must be unique within the catalog definition file.

3. In the Initial Context Factory field, specify the name of a Java class that implements the `javax.naming.InitialContextFactory` interface and generates the folder based on a set of parameters.

   For more information, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

4. In the `Visible` field, you can specify an EL value to conditionally display this folder to specific users or groups. This attribute takes an EL value. The default value is `#{true}`, which means that this resource is displayed in the catalog at all times.

5. Select **Insert Folder Contents** to display the contents of the custom folder directly in the catalog rather than displaying them under the folder.

6. Optionally, in the Custom Folder Attributes section, you can define the following attributes by choosing them from the **Add** menu:

   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

   Selecting an option on the Add menu creates a new row in the Catalog Attributes table. You can use this row to specify the name, type, and value for the new attribute.

7. Use the Custom Folder Parameters section to add parameters that were defined while implementing the factory class. You can bind these parameters to other artifacts in the application. Click the **Add** icon in the Parameters section to create a new row in the Parameters table.

   You may see a dropdown menu on the Add icon if the factory class exposes parameters, for example, Content Presenter-specific factory classes provide a `templateView` parameter.

16.2.3.6 Adding a Custom Content Provider

A custom content provider dynamically generates zero or more catalog entries at runtime. Catalog entries include folders, links, custom folders, components, and so on.

When you add a Custom Folder, a folder is created in the catalog and is displayed at runtime even if it is empty. However, when you add a Custom Content element, a folder is created in the catalog, but is displayed at runtime only if it has components inside it. Except for this difference, a custom content provider is similar to a custom folder.

To add a custom content provider:

1. From the **Add** menu in the Catalog section, select **Custom Content**.

   A `customContent` element is added to the catalog. The right side of the page displays fields to define the folder's attributes and parameters.

2. The `Id` field is populated with the value `customFolder` by default. You can specify a different value, but the ID must be unique within the catalog definition file.

3. In the Content Provider field, specify the fully qualified name of a class that implements the interface `oracle.adf.rc.spi.plugin.catalog.CustomContentProviderV2`.

   You can use the Browse icon to see a list of classes that implement this interface. Also, see Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog."

4. In the `Visible` field, you can specify an EL value to conditionally display this folder to specific users or groups. This attribute takes an EL value. The default value is `#{true}`, which means that this resource is displayed in the catalog at all times.

5. Optionally, in the Component Attributes section, you can define the following attributes by choosing them from the **Add** menu:

   - **Title**: A label to be used for the resource in the catalog. This attribute is defined by default. You can change its value, if required.
   - **Description**: A description of this resource.
   - **Subject**: Keywords to facilitate keyword searching of the Resource Catalog.
   - **ToolTip**: The tool tip to be displayed when you move the mouse over this resource name in the catalog.
   - **IconURI**: The icon to be displayed next to the resource name in the catalog.

   Selecting an option on the Add menu creates a new row in the Catalog Attributes table. You can use this row to specify the name, type, and value for the new attribute.

6. Use the Custom Content Parameters section to define parameters on the folder. The parameters you define here are passed to and interpreted by the `CustomContentProviderV2` implementation. Click the **Add** icon in the Parameters section to create a new row in the Parameters table.

16.2.3.7 Dragging and Dropping a Resource into the Catalog

In addition to the resources mentioned in the previous sections, you can add the following resources by dragging them and dropping them onto a selected folder in the catalog definition file:

- Portlets, from the Application Resources pane or the Resource Palette.
- Content from the Oracle WebCenter Portal's Content Service for Oracle WebCenter Content, from the Application Resources pane or the Resource Palette.

- Task flows, from the Application Resources pane or the Resource Palette.

  Note: If you add an ADF task flow to the catalog and want to export the catalog to an already deployed application, then you must deploy the task flow as a shared library to the deployed application.

  For more information, see Section 13.3, "Adding Custom Task Flows to Your Portal."

- Another catalog definition file, from the Application Navigator.

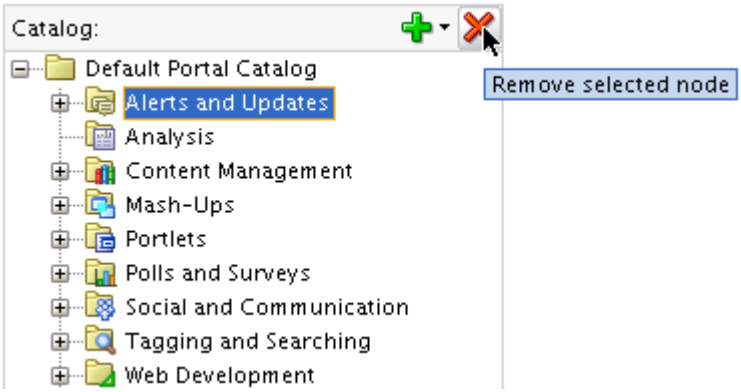
16.2.3.8 Editing a Resource

To edit a resource, select it in the Catalog section of the catalog definition file. The resource's attributes and parameters are displayed in the right half of the page. You can edit values for these attributes and parameters and save the catalog definition file. For details about a resource's attributes and parameters, see the section about adding that resource to the catalog.

16.2.3.9 Deleting a Resource

To delete a resource, select it in the Catalog section of the catalog definition file and click the **Remove selected node** icon, shown in Figure 16–6.

Figure 16–6 Delete Option in the Resource Catalog



16.2.3.10 Rearranging Resources in the Catalog

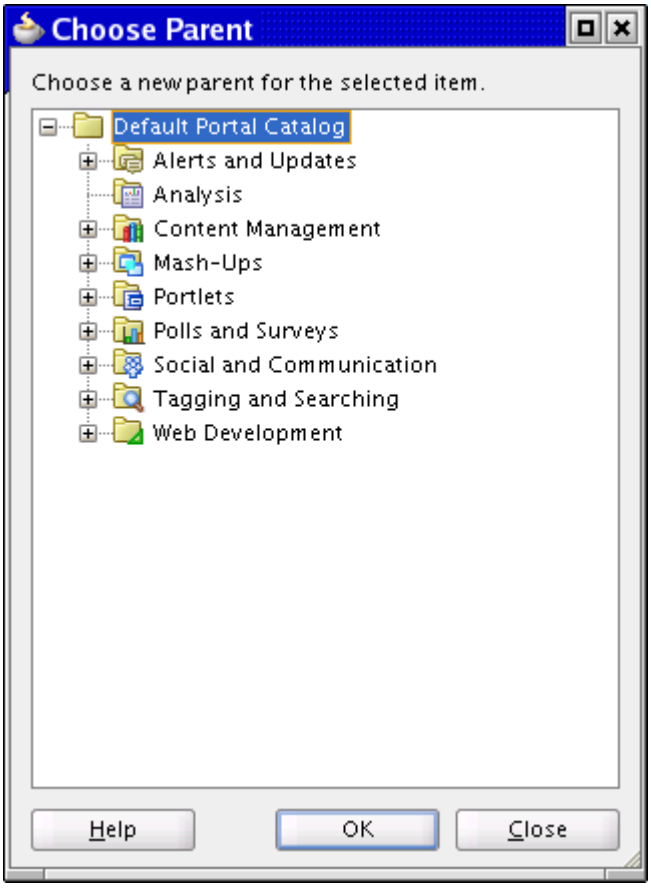
While organizing resources in the catalog, you can move a resource to a different folder in the hierarchy. You can rearrange resources in the following ways:

- Drag and drop them from one folder into another.
- Use the Change Parent context menu option to move a resource to a different folder.

To rearrange a resource using the Change Parent option:

1. Right-click the resource in the Catalog section of the catalog definition file and select **Change Parent** from the context menu.
2. In the Change Parent dialog, shown in Figure 16–7, select the folder in which to include the resource.

Figure 16–7 Change Parent Dialog



3. Click **OK**.

> **Note:** The OK button is enabled only when you select a valid location for the resource.

16.2.4 How to Expose Data Controls Created at Design Time

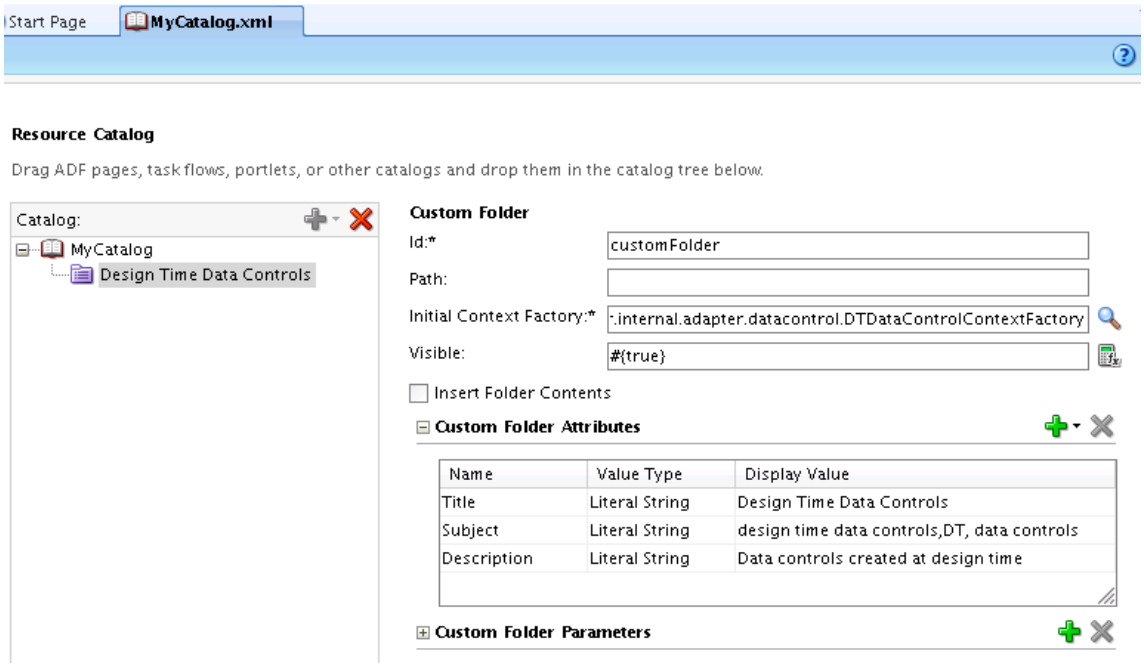
Oracle WebCenter Portal: Framework exposes a large number of service data controls that can be added to application pages and task flows. In addition, you may have created SQL or Web Services data controls in JDeveloper. To enable users to add these data controls to pages and task flows, you must expose the data controls in the Resource Catalog. For this, you must add a `customFolder` entry in your catalog definition file, as shown in the following example and Figure 16–8:

```
<customFolder id="dtDataControls"

factoryClass="oracle.webcenter.datacomposer.internal.adapter.datacontrol.DTDataCon
trolContextFactory">
  <attributes>
    <attribute value="Design Time Data Controls" attributeId="Title"/>
    <attribute value="Data controls created at design time"
attributeId="Description"/>
    <attribute value="design time data controls,DT, data controls"
attributeId="Subject"/>
    <attribute value="/adf/webcenter/folderlists_qualifier.png"
attributeId="IconURI"/>
```

```
  </attributes>
</customFolder>
```

Figure 16–8 Design Time Data Controls Folder in the Catalog Definition File Overview



To expose the `customFolder` as a translatable resource in the catalog, the folder's string attributes must be stored as resource strings in a resource bundle. For this, the catalog definition file must have a `resourceBundle` attribute and each translatable attribute must have an `isKey` attribute. Then the attribute value is saved in a resource bundle, as shown in the following example:

The resourceBundle attribute on the catalog definition:

```
<catalogDefinition id="DefaultCatalog" visible="#{true}"

resourceBundle="oracle.webcenter.portalapp.catalogs.DefaultCatalogBundle"
                  definitionFilter="portal.catalog.DefaultCatalogFilter"
                  xmlns="http://xmlns.oracle.com/adf/rcs/catalog">
```

The `isKey` attribute defined on the attributes of a translatable resource:

```
<customFolder id="dtDataControls"

factoryClass="oracle.webcenter.datacomposer.internal.adapter.datacontrol.DTDataCon
trolContextFactory">
  <attributes>
    <attribute value="DATACONTROL_FOLDER.TITLE" isKey="true" attributeId="Title"/>
    <attribute value="DATACONTROL_FOLDER.DESCRIPTION" attributeId="Description"
isKey="true"/>
    <attribute value="DATACONTROL_FOLDER.KEYWORDS" attributeId="Subject"
isKey="true"/>
    <attribute value="/adf/webcenter/folderlists_qualifier.png" isKey="false"
attributeId="IconURI"/>
  </attributes>
</customFolder>
```

For more information about using resource bundles, see Section 22.12, "Configuring Runtime Resource String Editing."

At runtime, the catalog displays a folder titled `Design Time Data Controls` irrespective of whether the application has any design-time data controls.

In a deployed application, the Resource Manager allows you to add design-time data controls individually to custom catalogs. Out-of-the-box catalogs cannot be edited at runtime. For more information, see "Adding Resources" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Exposing Custom Data Controls in an Already Deployed Application

If you created SQL or Web Services data controls in your application, and want to expose them in an already deployed application, you must perform the following steps:

- Package the application into an ADF shared library.
- Deploy the application to the WebLogic Server instance. For more information, see Section 56.2.3, "Rebuilding the Spaces Shared Library List."

When you perform this configuration, the deployed data controls are available in the Design Time Data Controls folder in the runtime Resource Registry. Users can add the data controls from the registry to any custom Resource Catalog, from where they can be consumed in pages and task flows.

16.2.5 How to Expose Data Controls Created at Runtime

Users can create data controls at runtime in the Resource Manager. If you want your Resource Catalog to dynamically display data controls created at runtime, add a `customFolder` entry in your catalog definition file, as shown in the following example:

```
<customFolder id="rtDataControls"

factoryClass="oracle.webcenter.datacomposer.internal.adapter.datacontrol.DataControlContextFactory">
  <attributes>
    <attribute value="Runtime Data Controls" attributeId="Title"/>
    <attribute value="Data controls created at runtime"
attributeId="Description"/>
    <attribute value="runtime data controls,RT,data controls"
attributeId="Subject"/>
    <attribute value="/adf/webcenter/folderlists_qualifier.png"
attributeId="IconURI"/>
  </attributes>
</customFolder>
```

To expose this folder as a translatable resource, the folder's string attributes must be stored as resource strings in a resource bundle. See Section 16.2.4, "How to Expose Data Controls Created at Design Time" for an example of how to create translatable resources.

At runtime, the catalog displays a folder titled `Runtime Data Controls` irrespective of whether the folder has any data controls.

16.2.6 How to Control Visibility of Portlets in the Resource Catalog

Out-of-the-box, a Resource Catalog displays portlets from all the available producers irrespective of whether the portlets are secured or not. As a result, users are allowed to

consume even secured portlets to which they do not have access. When a user adds such a portlet to a page, the portlet component is added, but it does not display any content. To ensure that users view and consume only those portlets that they can access, you can control the visibility of portlets in the Resource Catalog. Implement the `PortletItemSecurityHelper` interface to display portlets conditionally in the Resource Catalog, based on the user's permissions.

To control portlet visibility:

1. Implement the `PortletItemSecurityHelper` interface in your own item security helper class, for example, `CustomPortletSecurityHelper.java`.

   Example 16–1 shows a sample implementation of the `PortletItemSecurityHelper` interface. The following assumptions are made for the purpose of this example:

   - Grants are defined for each portlet to control who can see it.
   - A permission class, `PortletVisibilityPermission`, is defined to check a user's permissions on the portlet.

Example 16–1 Sample Implementation of PortletItemSecurityHelper

```
public class CustomPortletSecurityHelper extends PortletItemSecurityHelper
{
    public boolean canView(String producerId, String portletId)
    {
        String permissionPath = producerId + '/' + portletId;
        PortletVisibilityPermission permission = new PortletVisibilityPermission
(permissionPath, VIEW_ACTION);
        try
        {
          // throws AccessControlException if the permission is not granted
          JpsAuth.checkPermission(permission);
          canView = true;
          sLogger.logp(Level.FINE, SOURCE_CLASS, method,
                       "PortletVisibilityPermission check passed. Producer={0},
Portlet={1} Action=view",
                       new Object[]{producerId, portletId});
        }
        catch(AccessControlException ace)
        {
          canView = false;
          sLogger.logp(Level.FINE, SOURCE_CLASS, method,
                       "PortletVisibilityPermission check passed. Producer={0},
Portlet={1} Action=view",
                       new Object[]{producerId, portletId});
        }
        return canView;
    }
}
```

2. Create an `rc_ext.xml` file in the application's `META-INF` directory, and add the following code to register the implementation with the Resource Catalog:

```
<extension xmlns:="http://xmlns.oracle.com/adf/rc/extension"
           id="your.extension.id"
           name="Your Extension Name"
           version="11.1.1.4.0">
  <item-security-helper class="your.implementation.class.name"/>
</extension>
```

At runtime, when a user clicks Add Content in Composer, the Resource Catalog displays only those portlets on which the user has permissions.

16.2.7 How to Configure the Default Resource Catalog for an Application or Page

You can configure the default catalog at the application and page levels in the following ways:

- Registering it in the application's `adf-config.xml` file. This is an application-wide setting. For more information, see "Registering the Catalog in the adf-config.xml File".
- If you have multiple catalogs, by adding the `CatalogSelector`, which includes the logic to determine which catalog is shown for a specific page or locale. For more information, see Section 16.5.1, "How to Configure Multiple Resource Catalogs."
- If your application page contains Composer components, then by specifying the catalog name in the `Catalog` attribute on the `Page Customizable` component. You can specify a static or EL value. For more information, see Section B.1.1, "Page Customizable."

Registering the Catalog in the adf-config.xml File

To register a Resource Catalog in the `adf-config.xml` file so that it is used as the default catalog for the application:

1. Open the application's `adf-config.xml` file.
2. Change the `catalog-name` attribute value in the `<rcv:rcv-config>` element and specify the custom catalog name, as shown in the following example:

```
<rcv:rcv-config>
  <rcv:default-catalog
catalog-name="/oracle/webcenter/portalapp/catalogs/myCustomCatalog.xml"/>
</rcv:rcv-config>
```

   The `catalog-name` attribute identifies the catalog to be used as the default one.

   The path you specify is relative to the Resource Catalog root directory, for example, if you saved your new catalog definition as `C:\JDeveloper\mywork\RCSampleApp\Portal\Public_Html\oracle\webcenter\portalapp\catalogs\myCustomCatalog.xml`, then the `catalog-name` value would be `/oracle/webcenter/portalapp/catalogs/myCustomCatalog.xml`. The name uses the `/` separator because it represents a part of an MDS path.

   Note: For information about the Resource Catalog-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

3. Save the `adf-config.xml` file.

16.2.8 What You May Need to Know When Defining a Resource Catalog

To correctly implement Resource Catalogs, you must know where the application looks for the definitions and the XML schema upon which the definitions are based. See Appendix B, "Composer Component Properties and Files" for further information.

16.2.9 How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows

At runtime, when you add a task flow from the Resource Catalog to your application page in Composer, the task flow is automatically enclosed in a `Show Detail Frame` component. You can initialize attributes, such as the title, height, or width, associated with the enclosing `Show Detail Frame` by defining those attributes in the catalog definition file that contains the task flow. To differentiate the task flow's properties from the enclosing `Show Detail Frame`'s properties, you must use **`attr.`** as a prefix for `Show Detail Frame` properties.

For example, if a user adds a task flow to a page with a dark background setting, the task flow continues to use the background setting of the enclosing `Show Detail Frame` component. This may not be the look the user wants. To enable the user to change the `Show Detail Frame`'s background property, you must initialize and expose the background attribute in the Resource Catalog. At runtime, while editing the task flow's properties, a user can also change the `Show Detail Frame`'s background attribute.

Similarly, you can also initialize task flow parameters. You can define such parameters in the catalog definition file using an EL expression. At runtime, while editing the task flow's properties, a user can also change values for the exposed parameters. To differentiate the task flow's properties from the initialized parameters, you must use **`parameter.`** as a prefix for such parameters.

To initialize task flow parameters and `Show Detail Frame` properties:

1. Open the catalog definition file that contains the task flow.
2. Within the task flow's `<url>` or `<resource>` element, add an `<attribute>` entry for each parameter and `Show Detail Frame` attribute you want to define.

   The `attributeId` for a parameter must contain the parameter name along with a prefix of **`parameter.`**; the `attributeId` for a `Show Detail Frame` attribute must contain the attribute name along with a prefix of **`attr.`** as shown in the following example:

```
<url visible="#{true}"
factoryClass="oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourc
eFactory"
     id="wallMiniView"

url="taskflow://oracle/webcenter/peopleconnections/wall/controller/taskflows/Wa
llViewer.xml#WallViewer">
  <attributes>
    <attribute value="WALL_MINIVIEW.TITLE" attributeId="Title" isKey="true"/>
    <attribute value="WALL_MINIVIEW.DESCRIPTION" attributeId="Description"
isKey="true"/>
    <attribute value="#{securityContext.userName}"
attributeId="parameter.userName"/>
    <attribute value="false" attributeId="attr.stretchContent"/>
    <attribute value="never" attributeId="attr.showResizer"/>
  </attributes>
  <contents/>
</url>
```

> **Note:** Include the `isKey` attribute and set it to `true` only if the value is coming from a resource bundle. If not included, the `isKey` value defaults to `false`.

3. Save the catalog definition file.

16.3 Managing a Resource Catalog and Enabling Runtime Administration

The Resource Manager in a Framework application enables users with administrative privileges to create and modify application resources at runtime. By exposing catalogs in the Resource Manager, you allow administrators to manage these catalogs at runtime. You can also export selected catalogs from JDeveloper so that administrators can modify or delete them at runtime

This section outlines the tasks you can perform in JDeveloper to enable runtime administration of your catalogs. It contains the following sections:

- Section 16.3.1, "Managing Resource Manager Entries"
- Section 16.3.2, "Enabling Further Development of Deployed Resource Catalogs"
- Section 16.3.3, "Working with Spaces Resource Catalogs"

16.3.1 Managing Resource Manager Entries

In JDeveloper, use the Resource Manager-specific options on the context menu on a catalog definition file as follows:

- Create Portal Resource: Creates an entry for the catalog in the Resource Manager. The catalog can then be selected for use during page, template, and task flow creation.

  You must set certain attributes while creating an entry. Optionally, you can specify who can revise and delete the selected catalog in the Resource Manager. For more information, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

- Update Portal Resource: Edit the attributes that you defined while creating Resource Manager entries. For more information, see Section 17.2.3, "How to Update Portal Resource Properties."

- Remove Portal Resource: Remove the entry for this catalog from the Resource Manager registry. For more information, see Section 17.2.4, "How to Remove a Resource from the Resource Manager."

16.3.2 Enabling Further Development of Deployed Resource Catalogs

Out-of-the-box Resource Catalogs cannot be modified or deleted at runtime. However, users with administrator privileges can make a copy of an out-of-the-box Resource Catalog and edit its properties or delete it.

> **Note:** If you create a resource catalog at runtime and do not import it back into the design time environment, your new resource catalog may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The Edit Source option in the Resource Manager enables users to modify the content of a copied or custom Resource Catalog at runtime. However, as text editing is easier in JDeveloper than in the runtime source editor, users may choose to download a Resource Catalog, edit it in JDeveloper, and upload it back to the deployed application. At design time, this type of round-trip development includes importing the downloaded Resource Catalog, editing it in JDeveloper, and exporting it so that it can be uploaded back to the deployed application.

This section describes how to import and export Resource Catalogs in JDeveloper. For information about modifying catalog content, see Section 16.2.3, "How to Manage Content in the Resource Catalog."

16.3.2.1 Exporting a Resource Catalog

To expose a Resource Catalog in an already deployed application, you must export it from JDeveloper, then upload to the application using the runtime Resource Manager. When you export a catalog in JDeveloper, an EAR file is created with the catalog definition file and the `generic-site-resources.xml` file. The `generic-site-resources.xml` file contains metadata that helps identify the type of resource, a Resource Catalog in this case, so that the resource is displayed under the correct category in the Resource Manager.

For generic information, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

16.3.2.2 Importing a Resource Catalog

When you expose a Resource Catalog at runtime, users with privileges can add or delete components to suit their business requirements. However, to perform more complex tasks, such as applying catalog filters or adding custom resources that are not available in the deployed application, the runtime catalog must be downloaded, imported into JDeveloper and edited, exported from JDeveloper, and uploaded back to the runtime application.

> **Note:** To expose custom resources, such as task flows created in JDeveloper, that are not available in the deployed application, along with exporting the catalog you must also deploy the resource as a shared library.

When a user downloads a catalog from a deployed application, an EAR file is created containing the catalog definition XML file and the `generic-site-resources.xml` file. The `generic-site-resources.xml` file contains metadata specific to the Resource Catalog. This helps in identifying the type of resource, a Resource Catalog in this case. To edit the Resource Catalog in JDeveloper, you must first import the EAR file. On importing, the catalog definition file is extracted to the `siteResources` folder.

If the resource was downloaded from WebCenter Portal: Spaces, it is extracted to *`APPLICATION_ROOT`*`/oracle/webcenter/siteresources/scopedMD/guid`, where `guid` is the GUID of the space from which the resource was downloaded. Otherwise, the imported resource is created in *`APPLICATION_ROOT`*`/oracle/webcenter/portalapp`. You can open the catalog definition file and edit it as you would any other catalog at design time. The updated catalog must then be exported to the runtime application.

For generic information, see Section 17.4.2, "How to Import a Portal Resource into JDeveloper."

For information about downloading a catalog from the deployed application, see "Uploading and Downloading a Resource Catalog" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

16.3.3 Working with Spaces Resource Catalogs

Out-of-the-box, WebCenter Portal: Spaces provides several default Resource Catalogs for people to use. If these default catalogs are not suitable, you can build catalogs of your own within WebCenter Portal: Spaces. For details, see the chapter "Working with the Resources that Compose a Portal or Community" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

If you want to create a Resource Catalog that is beyond the editing capabilities of WebCenter Portal: Spaces, you can create the catalog in JDeveloper and then upload it to WebCenter Portal: Spaces. You do not have to develop the catalog from scratch, you can export an existing catalog from WebCenter Portal: Spaces and edit it in JDeveloper, or use one of the out-of-the-box catalogs as your starting point.

Oracle WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources, such as Resource Catalogs, to WebCenter Portal: Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build custom catalogs and other types of resources for WebCenter Portal: Spaces. For details, see Section 56, "Extending the Spaces Application Using JDeveloper."

16.4 Filtering Items in the Resource Catalog

If you want to arrange catalog content so that it is essentially the same for all users, but displays different subsets of the content to different users, then you can configure your Resource Catalog to filter out selected items based on specific criteria.

For example, if two users `A` and `B` have different privileges on a page, you can configure your application to display the entire catalog to user `A` and only a subset of items in the catalog to user `B`.

You can filter resources in the catalog either at the catalog level or at the individual resource level. This section explains both procedures. It contains the following sections:

- Section 16.4.1, "Resource-Level Filtering"
- Section 16.4.2, "Catalog-Level Filtering"

16.4.1 Resource-Level Filtering

You can display a resource conditionally by specifying an EL value for the `Visible` attribute on the resource. For information about editing the Visible attribute, see Section 16.2.3, "How to Manage Content in the Resource Catalog."

16.4.2 Catalog-Level Filtering

To apply a filter on the catalog, you must implement the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter` interface. Additionally, you must associate this filter with the Resource Catalog.

To implement the `CatalogDefinitionFilter`:

1. From the **File** menu, choose **New**.

2. In the New Gallery dialog, expand **General**, select **Java**, and then **Java Class**, and click **OK**.
3. In the Create Java Class dialog, specify a name for the class, for example, `CatalogFilter`.
4. Under the Optional Attributes section, add the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter` interface.
5. Click **OK**.
6. Specify the logic for filtering items.

   Note: When implementing the filter logic it is common to want information about the current user and current page. This information can be obtained from the `FacesContext`.

   Example 16–2 shows a sample `CatalogDefinitionFilter` implementation. It is configured to hide the `Discussions` folder (`forumsFolder`) for user `jdoe` from the custom catalog, `users-catalog.xml`, described in Section 16.2, "Creating a Custom Resource Catalog." The entire catalog, with the `ADF Faces Components` and `Discussions` folders, is shown to all other users.

Example 16–2 Sample Showing CatalogDefinitionFilter Implementation

```
package webcenter;

import oracle.adf.rc.catalog.CatalogElement;
import oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter;

import java.util.Hashtable;

import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;

public class CatalogFilter implements CatalogDefinitionFilter {

  public boolean includeInCatalog(CatalogElement element, Hashtable env) {

    ExternalContext ectx =
FacesContext.getCurrentInstance().getExternalContext();

    if ("jdoe".equals(ectx.getRemoteUser()) &&
"forumsFolder".equals(element.getId()))
      return false;
    else
      return true;
  }

}
```

7. Open the catalog definition file.
8. Select the root node in the Catalog section, and set the Catalog Filter field to the name of the Java class you created, for example, `webcenter.CatalogFilter`.

   This adds a `definitionFilter` attribute in Source view of the file.
9. Save the catalog definition file.

16.5 Configuring Multiple Resource Catalogs

You can expose different catalogs to different users based on specific criteria, such as the page being edited, the user, or the user role. This section describes how to associate multiple Resource Catalogs with your application. It contains the following subsections:

- Section 16.5.1, "How to Configure Multiple Resource Catalogs"
- Section 16.5.2, "What Happens at Runtime"

16.5.1 How to Configure Multiple Resource Catalogs

To use multiple catalogs, you must implement the `ResourceCatalogSelector` API to select the appropriate catalog for the user and the page being edited. You must include the `catalog-selector` entry in your application's `adf-config.xml` file to specify the name of your `ResourceCatalogSelector` class. If you do not specify a `catalog-selector`, the default catalog is used for all editable pages and all users see the same content. If you specify a `catalog-selector`, the default catalog is used only when the specified selector returns null.

To implement multiple Resource Catalogs:

1. Create your custom catalogs by performing the steps outlined in Section 16.2, "Creating a Custom Resource Catalog."
2. Implement the `oracle.adf.rc.model.config.ResourceCatalogSelector` interface in your own Resource Catalog selector class, for example, `CatalogSelector.java`.

   Example 16–3 shows a catalog selector implementation where a custom Resource Catalog, `admin-catalog.xml`, is shown to `ahunold` who has administrator privileges, and `users-catalog.xml` is shown to users `sking` and `jdoe`. If there is a problem with rendering either of these catalogs, then the default catalog is shown to users.

Example 16–3 Sample ResourceCatalogSelector Showing Entries for Multiple Resource Catalogs

```
package webcenter;


import javax.faces.context.FacesContext;

import java.util.Map;

import javax.faces.component.UIViewRoot;
import javax.faces.context.FacesContext;
import javax.faces.context.ExternalContext;

import oracle.adf.rc.model.config.ResourceCatalogSelector;
import oracle.adf.rc.model.dc.RCVContext;

import oracle.adf.rc.model.config.ResourceCatalogSelector;

/**
 * CatalogSelector is a sample implementation of the ResourceCatalogSelector
 * interface.
 *
```

```
 * This implementation is based on the authenticated user
 */
public class CatalogSelector implements ResourceCatalogSelector {
  public CatalogSelector() {
  }

  /**
   * Returns the Id of the catalog that must be used for the current
   * user.
   *
   * @param context a Map cont
   * @return id of the catalog to be used for the current user or
   *         null if the default catalog must be used.
   */
  public String getCatalogName(Map context) {
    // if no catalog is selected, return null & ensure that the Composer uses the
    // default catalog defined in the rcv-config
    String retval = null;

    // get information about the current user.
    FacesContext fctx = FacesContext.getCurrentInstance();
    ExternalContext ectx = fctx.getExternalContext();
    String user = ectx.getRemoteUser();

    if (user == null || user.length() == 0) {
      // user name is not available so use the default catalog
      retval = null;

    }
    else if (user.equals("jdoe") || user.equals("sking")  ) {
      // return users-catalog for users jdoe and sking
      retval = "users-catalog";


    }
    else if (user.equals("ahunold")) {
      // return admin-catalog for user ahunold
      retval = "admin-catalog";

    }

    return retval;

    }
}
```

3. In your `adf-config.xml` file, enter the name of the class you created for selecting a Resource Catalog with the `<rcv:catalog-selector>` tag as show in the following example:

```
<rcv:rcv-config>
<rcv:default-catalog
catalog-name="/oracle/webcenter/portalapp/catalogs/default-catalog.xml"/>
<rcv:catalog-selector class-name="webcenter.CatalogSelector"/>
</rcv:rcv-config>
```

   The value for `catalog-name` corresponds to the default catalog definition file with the MDS path.

Note: For information about the Resource Catalog-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

16.5.2 What Happens at Runtime

Different catalogs are displayed to users based on the criteria you specified.

Based on the `ResourceCatalogSelector` implementation in Example 16–3, Figure 16–9 and Figure 16–10 show the two catalogs that are shown to user `sking` or `jdoe` and to administrator `ahunold` respectively.

Figure 16–9 Catalog Displayed to sking and jdoe

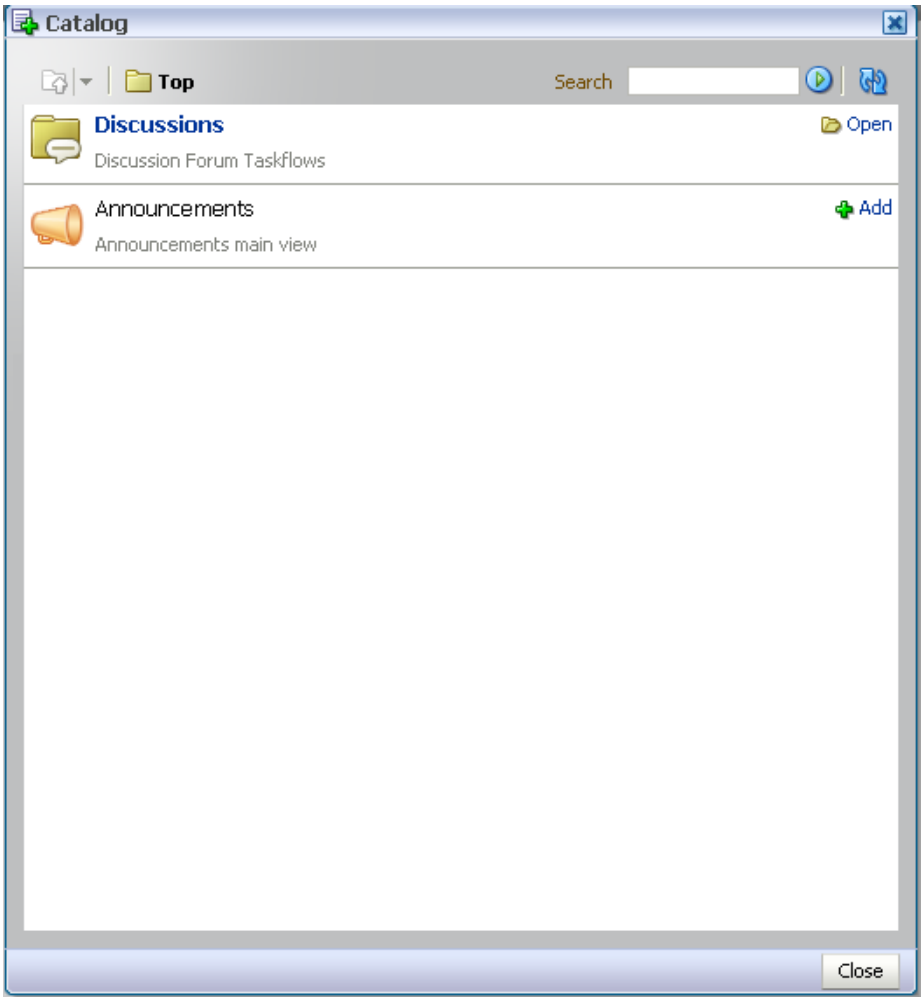
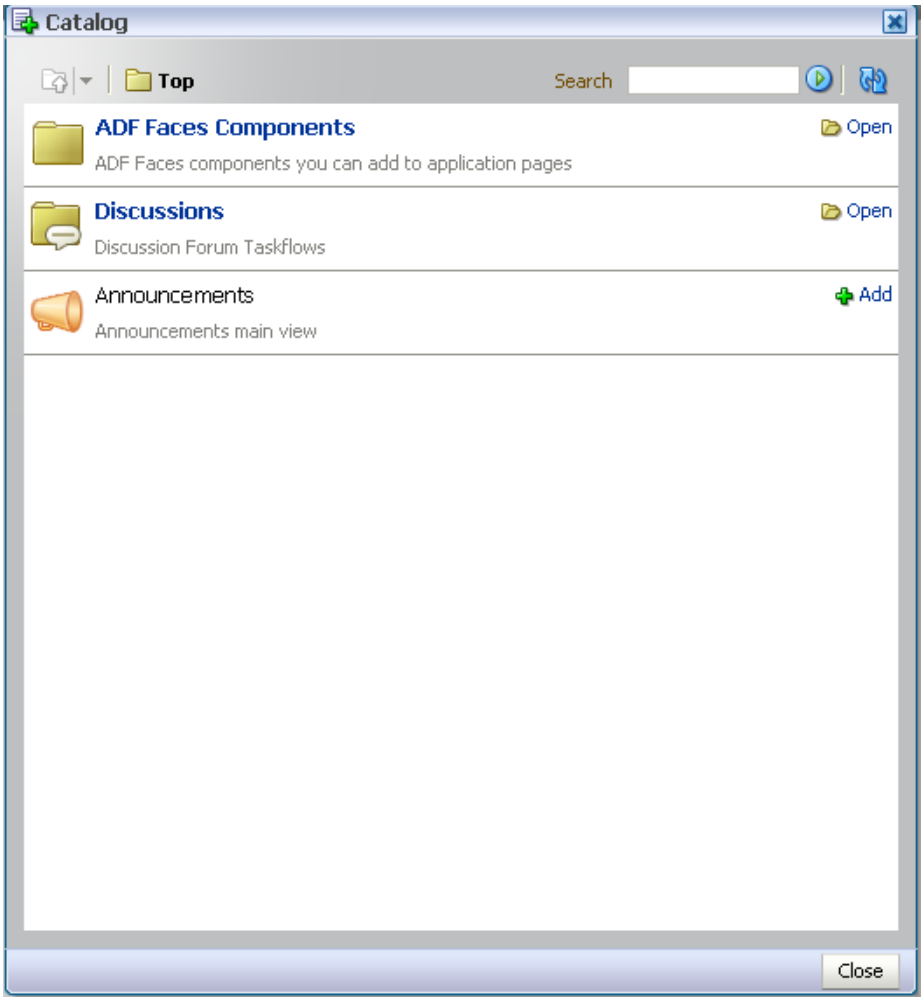


Figure 16–10 Resource Catalog Displayed to the ahunold



16.6 Troubleshooting Problems with the Resource Catalog

This section provides information to assist you in troubleshooting problems you may encounter while using the Resource Catalog.

For information about configuring logging, see "Configuring ADF Logging for Composer".

Problem

Resource Catalog is empty.

Solution

The default catalog is not available to MDS. Ensure that the deployment profile contains the necessary entries to copy the default catalog file. Also, ensure that in the MDS section of `adf-config.xml`, a namespace entry points to the default catalog file.

Problem

A project task flow does not appear in the Resource Catalog.

Solution

The task flow must be packaged as an ADF Library (JAR file) and added to the project. You must also ensure that the task flow ID is given in the following format:

```
taskflow://Path_to_Task_Flow/Task_Flow_Definition_File_Name#Task_Flow_ID
Ex.
taskflow://oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer
.xml#WallViewer
```

17

Adding and Using the Resource Manager

This chapter explains how to enable users to administer portal resources at runtime using the Resource Manager.

Using the Resource Manager users can also download runtime portal resources (from WebCenter Portal: Framework applications or WebCenter Portal: Spaces) and import them into Oracle JDeveloper for further development. These resources can then be exported from JDeveloper and uploaded back into the deployed application. Portal resources can also be created from scratch in JDeveloper and uploaded to the deployed application.

This chapter includes the following subsections:

- Section 17.1, "Introduction to Runtime Resource Management"
- Section 17.2, "Working with Portal Resources in JDeveloper"
- Section 17.3, "Enabling Runtime Administration of Your Portal"
- Section 17.4, "Enabling Round-Trip Development of Resources"

For information about using the Resource Manager to administer portal resources at runtime, see the chapter "Working with the Resources that Compose a Portal or Community" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

17.1 Introduction to Runtime Resource Management

A portal is a constantly evolving application. While the initial framework for the portal is designed prior to deployment, this really acts as a starting point. The design of the portal is likely to be enhanced by the actual users of the portal.

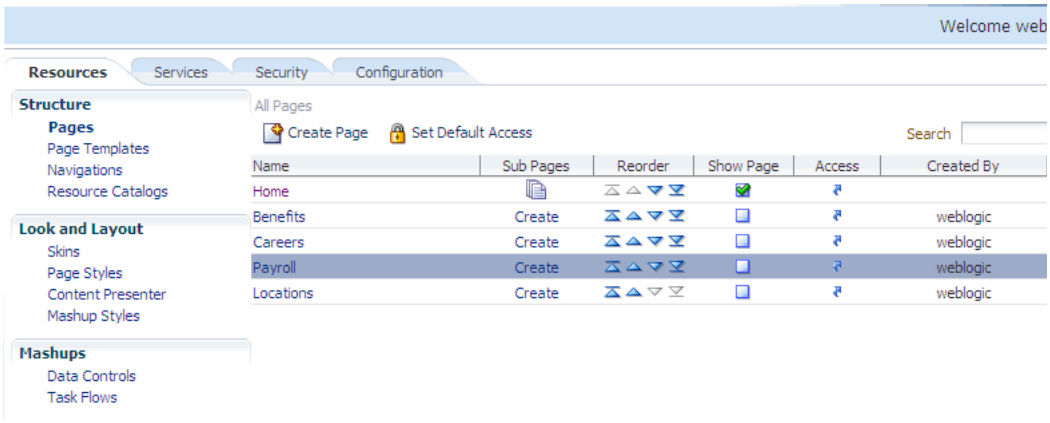
This runtime resource management provides:

- **Runtime development**—Continue developing your portal after deployment by creating resources, such as pages, skins, and data controls at runtime. For more information, see Section 17.3.2, "How to Use the Resource Manager at Runtime."
- **Round-trip development**—Pull resources from the deployed portal into JDeveloper for further development and then upload the enhanced resources back into the running application. For more information, see Section 17.4, "Enabling Round-Trip Development of Resources."
- **Iterative development**—Supply a running portal with new or improved visualizations of metadata-only resources from JDeveloper without the need to redeploy or restart the application. For more information, see Section 1.5, "Preparing for Iterative Development."

- **Extension of WebCenter Portal: Spaces**—Build and manage new resources for WebCenter Portal: Spaces in JDeveloper and package them on a periodic basis for distribution to the runtime portal. For more information, see Chapter 56, "Extending the Spaces Application Using JDeveloper."

WebCenter Portal enables runtime resource management through the Resource Manager. Using the Resource Manager, users with the appropriate privileges can continue developing the portal after the application has been deployed (Figure 17–1).

Figure 17–1 The Resource Manager in a WebCenter Portal: Framework Application



At runtime, users with the appropriate privileges can create and edit:

- Pages
- Page templates
- Navigation models
- Resource Catalogs
- Skins
- Page styles
- Content Presenter display templates
- Mashup styles
- Data controls
- Task flows

Using the Resource Manager, portal users can also download resources, or an entire application, from the runtime environment, edit them in JDeveloper, and then upload them back into the deployed application.

Users can also create resources from scratch in JDeveloper, export them as portal resources, and then upload them into the deployed portal. This is a very useful way of creating more fully functional portal resources for use in WebCenter Portal: Spaces.

This process is known as round-trip development. For more information, see Section 17.4, "Enabling Round-Trip Development of Resources."

17.2 Working with Portal Resources in JDeveloper

For a resource to be included in the Resource Manager and therefore available for runtime administration, it must be created as a portal resource.

To create a resource as a portal resource it must be located in the `oracle/webcenter/portalapp` directory of the application project.

This section includes the following subsections:

- Section 17.2.1, "How to Add a Resource to the Resource Manager"
- Section 17.2.2, "What You May Need to Know About the Content Directory"
- Section 17.2.3, "How to Update Portal Resource Properties"
- Section 17.2.4, "How to Remove a Resource from the Resource Manager"

17.2.1 How to Add a Resource to the Resource Manager

To enable runtime administration of a resource it must be included in the Resource Manager. To do this, you need to identify the resource as a portal resource.

To add a resource to the Resource Manager:

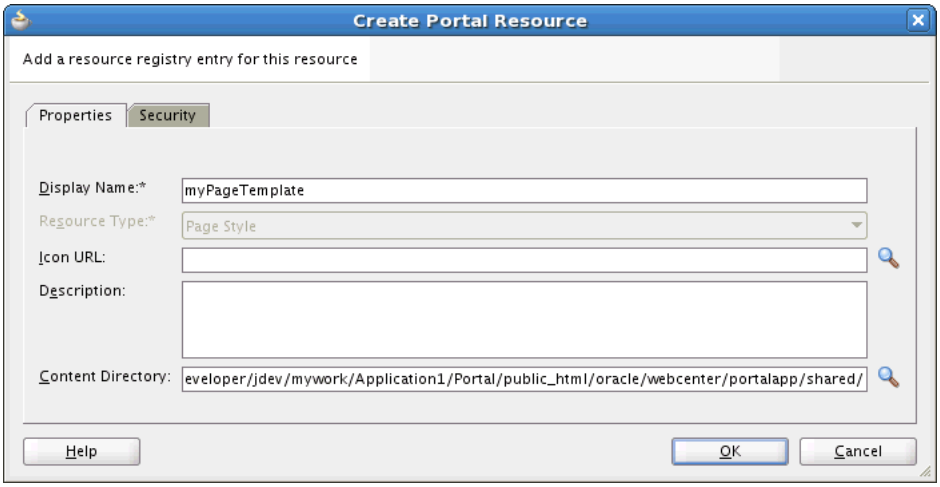
1. In the Application Navigator, right-click the resource that you want to include in the Resource Manager and choose **Create Portal Resource**.

   Note: If you do not see the **Create Portal Resource** option in the context menu, the resource is not under the `oracle/webcenter/portalapp` directory. To move the resource to the correct directory, right-click it, choose **Refactor**, and then **Move**.

2. In the Create Portal Resource dialog (Figure 17–2), in the **Display Name** field, edit the name if required.

   This is the name that is listed in the Resource Manager at runtime, so make sure that users are able to easily identify the resource from the display name.

Figure 17–2 The Create Portal Resource Dialog



3. The **Resource Type** field identifies the kind of resource you are adding to the Resource Manager. You cannot edit this field.

4. In the **Icon URL** field, enter the path and file name of an image to use as a graphical representation of the resource.

   For page styles, this icon is displayed in the Create Page dialog.

5. In the **Description** field, enter a description for the resource. This description should help users determine if they want to use the resource in their application.

   The description is displayed in the About dialog for the resource and in the Resource Manager at runtime.

6. In the **Content Directory** field, enter or browse for the directory where dependent objects for the resource, such as images, JavaScripts, style sheets, and HTML files, are stored. For more information, see Section 17.2.2, "What You May Need to Know About the Content Directory."

   Note: The folder specified here is included in the export archive when the resource is exported, so any dependent objects created under the folder are pushed into the target system.

   This field appears only for resources that can include such dependent objects.

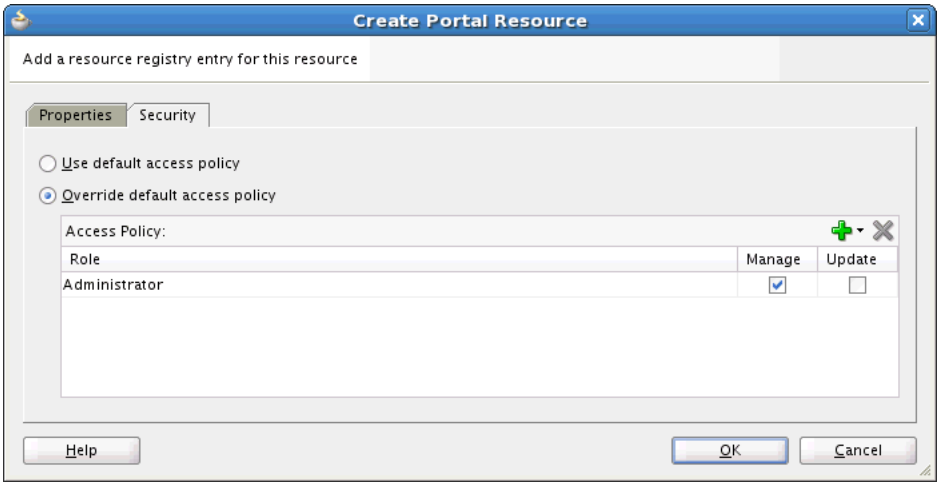
   ---

7. If you want to specify resource-level security for the resource, click the **Security** tab. Otherwise, skip to step 11.

   Resource-level security enables you to override the application-level security for individual resources to specify who can edit and delete the resource at runtime.

8. Select **Override default access policy** (Figure 17–3).

Figure 17–3 Create Portal Resource - Security Tab



9. Click the **Add permissions to new role** icon and select the role for which you want to grant permissions.

10. Select the **Manage** checkbox to grant full access to the resource at runtime. Users with the role can edit and delete the resource.

    Select the **Update** checkbox to grant the permission to edit the resource at runtime. Users with the role can edit the resource, but they cannot delete it.

Note: The user must also have the appropriate permissions to access the Resource Manager. For more information, see Section 17.3.1, "How to Add the Resource Manager Task Flow."

11. Click **OK**.

    The resource is added to the Resource Manager and users with the appropriate privileges can manage the resource at runtime.

17.2.2 What You May Need to Know About the Content Directory

If you are making a resource available for runtime management, any dependent objects referenced by the resource, such as images, JavaScripts, style sheets, or HTML files, must be stored under the following directory:

APPLICATION_ROOT/Portal/public_html/oracle/webcenter/portalapp/shared

In addition, as a best practice, it is recommended to store dependent objects in resource-specific directories under the `shared` directory. For example, images for a page template named `Template1` could be stored in:

APPLICATION_ROOT/Portal/public_
html/oracle/webcenter/portalapp/shared/pageTemplates/Template1

This avoids any potential conflict when additional resources that include dependent objects are imported from runtime.

To reference dependent objects in a resource, use the following format:

/oracle/webcenter/portalapp/shared/*object*

For example:

```
<af:image
source="/oracle/webcenter/portalapp/shared/pageTemplates/Template1/logo.gif"
id="pt_12">
```

When adding the resource as a portal resource (as described in Section 17.2.1, "How to Add a Resource to the Resource Manager"), you must ensure that the **Content Directory** points to the appropriate `shared` directory.

Note: For runtime-created resources, the default content directory is `/oracle/webcenter/siteresources/scopedMD/shared`.

17.2.3 How to Update Portal Resource Properties

You can update the details of resources in the Resource Manager.

To update portal resource properties:

1. In the Application Navigator, right-click the resource that you want to edit and choose **Update Portal Resource**.
2. In the Update Portal Resource dialog, edit the fields as necessary.

   For information about the fields in this dialog, see Section 17.2.1, "How to Add a Resource to the Resource Manager."
3. Click **OK**.

17.2.4 How to Remove a Resource from the Resource Manager

If you no longer want users to be able to administer a particular resource at runtime, you can remove it from the Resource Manager.

To remove a resource from the Resource Manager:

1. In the Application Navigator, right-click the resource that you want to remove and choose **Remove Portal Resource**.
2. In the Remove Portal Resource dialog, click **Yes** to confirm that you want to remove the resource from the Resource Manager.

   The resource is removed from the Resource Manager and can no longer be edited or downloaded at runtime. The resource itself is not deleted, so it is still available for use within the Framework application.

17.3 Enabling Runtime Administration of Your Portal

You can enable portal administrators to manage resources at runtime by adding the Resource Manager to a page in your portal.

If you use WebCenter Portal's Framework application template to create your application, an Administration page is automatically included in the application. This page includes a tab for the Resource Manager.

If you prefer, you can add the Resource Manager to any other page in your application by adding the List of Resource Types task flow to that page.

This section includes the following subsections:

- Section 17.3.1, "How to Add the Resource Manager Task Flow"
- Section 17.3.2, "How to Use the Resource Manager at Runtime"

17.3.1 How to Add the Resource Manager Task Flow

For applications created using WebCenter Portal's Framework application template, the Resource Manager task flow is available out of the box through the WebCenter Portal Administration Console (Resources tab). For details, see the section "Managing Application Resources" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

In addition, just like other task flows, you can add the Resource Manager task flow to your application pages. This might be especially useful if you are not using WebCenter Portal's Framework application template, and the WebCenter Portal Administration console is therefore not part of your project.

When you add the Resource Manager task flow to a page, anyone with view access to the page can view the task flow and perform view related operations on resources, such as viewing resource properties, previewing resources, and so on. For users to be able to perform other operations on resources, such as creating new resources, editing resources, or deleting resources, they must have the appropriate permissions on the type of resource they want to work with. These permissions can be provisioned using the Role Manager task flow. For more information, see Section 69.4, "Using the Role Manager Task Flow." By default, the `Administrator` role has manage access on all resource types exposed in the Resource Manager task flow.

To add the Resource Manager task flow:

1. Open the page to which you want to add the Resource Manager.

2. In the Resource Catalog, expand the **WebCenter Portal - Services Catalog** and then **Task Flows**.
3. Select the **List of Resource Types** task flow and drag it to the desired location on the page.
4. From the menu that displays, select **Region**.

   You may be prompted to add the Resource Manager library to the project. Confirm by clicking **Add Library**.
5. The Resource Manager task flow is added to the page.
6. To enable users to upload and download resources at runtime, the `usesUpload` property for the form that contains the Resource Manager task flow must be set to true:

```
<af:form id="f1" usesUpload="true">
```

17.3.2 How to Use the Resource Manager at Runtime

At runtime, the Resource Manager provides access to administration tools for the resources available to the portal. For example, the administrator can create new resources (navigation models, page templates, mashup styles, and so on), edit existing resources, upload resources exported from other applications, and so on.

Table 17–1 lists the types of resources that you can manage at runtime and provides links to the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* where you can find out more about working with these resources at runtime.

Table 17–1 WebCenter Portal Resources That Can Be Managed at Runtime

Resource	Description	Documentation
Pages	Create and manage pages for your application.	"Building Pages"
Page Templates	Define how individual pages and groups of pages display on a user's screen	"Working with Page Templates"
Navigation Models	Define how to link together information from multiple sources, such as spaces, pages, content repositories, and even external Web pages	"Working with Navigation"
Resource Catalogs	Define the components and connections that WebCenter Portal: Spaces users can add to their pages, page templates, and task flows.	"Working with Resource Catalogs"
Skins	Define the appearance and look and feel, including colors and fonts, of a specific space or the entire application.	"Working with Skins"
Page Styles	Define the layout of a newly created page and may also dictate the type of content the page supports.	"Working with Page Styles"
Content Presenter Display Templates	Manage the templates for presenting content.	"Publishing Content Using Content Presenter"

Table 17–1 (Cont.) WebCenter Portal Resources That Can Be Managed at Runtime

Resource	Description	Documentation
Mashup Styles	Manage the templates available for creating task flows	"Managing Mashup Styles"
Data Controls	Create and manage data controls, which connect to and read data from external repositories, particularly for use in business mashups.	"Creating and Managing Data Controls"
Task Flows	Create and manage task flows based on a selected mashup style.	"Creating and Managing Task Flows"

This section includes the following subsections:

- Section 17.3.2.1, "Accessing the Resource Manager"
- Section 17.3.2.2, "Creating a Resource"
- Section 17.3.2.3, "Uploading and Downloading a Resource"
- Section 17.3.2.4, "Editing a Resource"
- Section 17.3.2.5, "Copying a Resource"
- Section 17.3.2.6, "Setting Security for a Resource"
- Section 17.3.2.7, "Deleting a Resource"
- Section 17.3.2.8, "Showing or Hiding a Resource"
- Section 17.3.2.9, "Editing the Source Code of a Resource"
- Section 17.3.2.10, "Setting Properties on a Resource"
- Section 17.3.2.11, "Previewing a Resource"

Note: The process for working with pages at runtime is different from other resources. For more information, see the section "Working with Pages" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*

17.3.2.1 Accessing the Resource Manager

To access the Resource Manager at runtime, users can navigate to the built-in Administration page (if it is available in the application) and click the **Resources** tab. If the Administration page is not included in the application, you must direct users to the page that includes the Resource Manager.

17.3.2.2 Creating a Resource

Users may want to create custom resources at runtime if the resources available in the Framework application do not meet their requirements.

At runtime, users can create certain resources from scratch. There are some resources that users can create only by extending an existing resource. For example, at runtime, users can create a task flow from scratch, but can create a skin only by extending an existing skin. Other resources, such as Content Presenter display templates, cannot be created at runtime.

Table 17–2 describes the support available for creating resources at runtime.

Table 17–2 Support for Creating Resources at Runtime

Resource	Users Can Create the Resource at Runtime?	How Can the Resource Be Created?
Pages	Yes	From scratch and by creating a copy of an existing page
Page Templates	Yes	Only by creating a copy of an existing page template
Navigation Models	Yes	From scratch and by creating a copy of an existing navigation model
Resource Catalogs	Yes	From scratch and by creating a copy of an existing Resource Catalog
Skins	Yes	Only by creating a copy of an existing skin
Page Styles	Yes	Only by creating a copy of an existing page style and editing the source
Content Presenter Display Templates	No	NA
Mashup Styles	Yes	Only by creating a copy of an existing mashup style and editing the source
Data Controls	Yes	From scratch and by creating a copy of an existing data control
Task Flows	Yes	From scratch and by creating a copy of an existing task flow

To create a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left navigation panel, select the type of resource that you want to create.
3. On the menu bar, click **Create**.
4. In the resulting dialog, enter details for the resource.

   These details depend on the type of resource you are creating. For links to the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* where you can find out more about creating resources at runtime, see Table 17–1.
5. Click **Create**.

   The newly created resource is listed in the Resource Manager. The gray icon next to a resource indicates that it is not yet published an hence is not available to users for use. For information about publishing resources, see Section 17.3.2.8, "Showing or Hiding a Resource."

Note: If you create resources at runtime and do not import them back into the design time environment, your new resources may be lost when the application is redeployed. For more information, see the "Preserving Resource Customizations" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

17.3.2.3 Uploading and Downloading a Resource

The Upload and Download options enable post-deployment, round-trip application development. These actions greatly simplify the process of bringing new or revised resources from JDeveloper into your application and pushing them back into development from your application to JDeveloper as needed.

For more information, see Section 17.4, "Enabling Round-Trip Development of Resources."

17.3.2.4 Editing a Resource

Authorized users can edit Framework application resources using the Resource Manager, regardless of whether they were created at design time or runtime, or have been uploaded to the application after deployment.

To edit a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to edit.
3. In the right panel, select the resource that you want to edit.
4. From the menu bar, click **Edit** and then choose **Edit**.
5. In the resulting dialog, edit the details for the resource.

   These details depend on the type of resource you are editing. For links to the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* where you can find out more about editing resources at runtime, see Table 17–1.
6. Click **OK**.

17.3.2.5 Copying a Resource

At runtime, users can create a copy of a resource. This feature is useful for:

- Creating a backup of a resource
- Updating a resource while keeping the original in use
- Using an existing resource as the starting point for a new resource.

You can create a copy of all types of resources except pages and Content Presenter display templates. When you create a copy of a resource, the copy is marked as hidden.

To copy a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to copy.
3. In the right panel, select the resource that you want to copy.

4. From the menu bar, click **Edit** and then choose **Copy**.
5. In the Copy dialog, in the **Display Name** field, enter a name for the new resource.
6. In the **Description** field, enter a description of the new resource.
7. Click **Copy**.

17.3.2.6 Setting Security for a Resource

Users can use security settings to specify who can revise and remove a selected resource. By default, resource access is controlled by application-level permissions. The Security Settings dialog provides a way of overriding application-level permissions and defining specific permissions on a selected resource.

To set security for a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource for which you want to set security.
3. In the right panel, select the resource for which you want to set security.
4. From the menu bar, click **Edit** and then choose **Security Settings**.
5. In the Security Settings dialog, select **Use Custom Permissions** to define who may manage and update the selected resource.

   To inherit access settings from those defined for the application, select **Use Application Permissions**.
6. Click **Add Users** or **Add Groups** to select the user or group to whom you want to grant resource access.
7. Select the **Manage** checkbox to grant full access to the resource. Such users can edit and delete the resource.

   Select the **Update** checkbox to grant the permission to edit the resource. Such users can edit the resource, but they cannot delete it.
8. To revoke access, select the user or group and click **Delete Access**.
9. Click **Set Access**.

17.3.2.7 Deleting a Resource

When a resource is no longer required in an applications, users may delete it. Users can delete resources created at runtime or design time, and uploaded resources.

To delete a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to delete.
3. In the right panel, select the resource that you want to delete.
4. From the menu bar, click **Edit** and then choose **Delete**.
5. In the Delete confirmation dialog, click **Delete**.

17.3.2.8 Showing or Hiding a Resource

All resources, whether created at runtime or design time, can be marked as hidden or available. For all resources that are available for use, a green tick mark and the word

"Available" appear next to the resource's name in the Resource Manager. A gray icon and the word "Hidden" next to a resource's name indicate that the resource is marked as hidden and cannot be used in the application.

When you create or upload a resource, by default it is marked as hidden. A hidden resource is not available for use in the application. You can still use hidden resources in the Resource Manager, for example, to copy them or to use them as the starting point for another resource.

To show or hide a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to show or hide.
3. In the right panel, select the resource that you want to show or hide.
4. From the menu bar, click **Edit** and then choose **Show** or **Hide**, depending on whether the resource is current shown or hidden.

   If the resource was previously hidden, the **Show** option displays in the **Edit** menu. When you choose **Show**, the resource becomes available for use in the application (indicated by a green tick mark and the word "Available" next to the resource)

   If the resource was previously shown, the **Hide** option displays in the **Edit** menu. When you choose **Hide**, the resource becomes unavailable for use in the application (indicated by a gray icon and the word "Hidden" next to the resource).

17.3.2.9 Editing the Source Code of a Resource

To give you more control over resource editing at runtime, the Resource Manager provides the **Edit Source** option. You can use this option to edit the underlying source code of all your resources except data controls. The changes are saved to the MDS, and available immediately.

You may want to edit a resource's source code to make advanced edits to its code without having to having to go through the entire round-trip development cycle. For more information about round-trip development, see Section 17.4, "Enabling Round-Trip Development of Resources."

You can even use the **Edit Source** option to create a resource from scratch, by creating the resource and then replacing its default source code with your own original code. Note, however, that due to the heavy hand-coding requirement, this scenario is not recommended.

To edit the source code of a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource for which you want to edit the source code.
3. In the right panel, select the resource for which you want to edit the source code.
4. From the menu bar, click **Edit** and then choose **Edit Source**.
5. In the Edit Source dialog, edit the code as required.

   The XML syntax in the code is validated and an error message is displayed if you miss any tags or add them incorrectly. Validation is not performed for non-XML files, such as CSS files.
6. Click **Edit Source**.

17.3.2.10 Setting Properties on a Resource

Each type of resource has certain associated properties that define its display properties, availability, and attributes. Authorized users can edit these properties for resources created or uploaded at runtime by using the Edit Properties dialog. Users cannot edit properties for resources created at design time.

To set properties on a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource for which you want to set properties.
3. In the right panel, select the resource for which you want to set properties.
4. From the menu bar, click **Edit** and then choose **Edit Properties**.
5. In the **Display Name** field, edit the display name for the resource.
6. In the **Description** field, edit the description for the resource.
7. In the **URI Icon** field, enter a standard URI path to an icon to represent the resource, for example:

   ```
   /myCompany/webcenter/page/images/myimage.png
   ```

   You can either specify an absolute URL or a relative URL that points to an image located somewhere in the application.
8. In the **Category** field, enter a category name to help classify the resource within the application.
9. In the Attributes section, enter values for any custom attributes for the resource by entering the **Name** and an appropriate **Value** for the attribute. For information about the custom attributes that can be added for the different resources, see the appropriate chapter for the resource type as listed in Table 17–1.
10. Click **OK**.

17.3.2.11 Previewing a Resource

You can preview page templates, mashup styles, and task flows to check how they look while you are editing them.

To preview a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to preview.
3. In the right panel, select the resource that you want to preview.
4. From the menu bar, click **Preview**.
5. When you have finished previewing the resource, click **Close**.

17.4 Enabling Round-Trip Development of Resources

If you enable runtime administration of your portal resources, you need to be able to pull those resources back into JDeveloper. For example, if a portal user adds a new navigation model, you may want to further refine it. To do this, you need to download the resource from the deployed application, using the Resource Manager. You then import the resource into JDeveloper, carry out your development work, export the

resource, and finally upload it back into the deployed application. This is known as round-trip development.

You can do this with resources in any deployed WebCenter Portal application that includes the Resource Manager, including WebCenter Portal: Spaces.

Note: WebCenter Portal provides a special JDeveloper workspace (`DesignWebCenterSpaces.jws`) to help you develop and upload resources to WebCenter Portal: Spaces. Chapter 56, "Extending the Spaces Application Using JDeveloper" tells you where to find the sample workspace and explains how to use the `WebCenterSpacesResources` project to build resources for use in WebCenter Portal: Spaces.

The process for round-trip development works as follows:

1. Build Framework application in JDeveloper
2. Deploy application
3. Users add/edit resources and add content to the portal at runtime
4. Download resource to an EAR file using the Resource Manager.

   See Section 17.4.1, "How to Download a Resource Using the Resource Manager"
5. Import EAR file into JDeveloper

   See Section 17.4.2, "How to Import a Portal Resource into JDeveloper"
6. Edit resource in JDeveloper
7. Export resource to EAR file

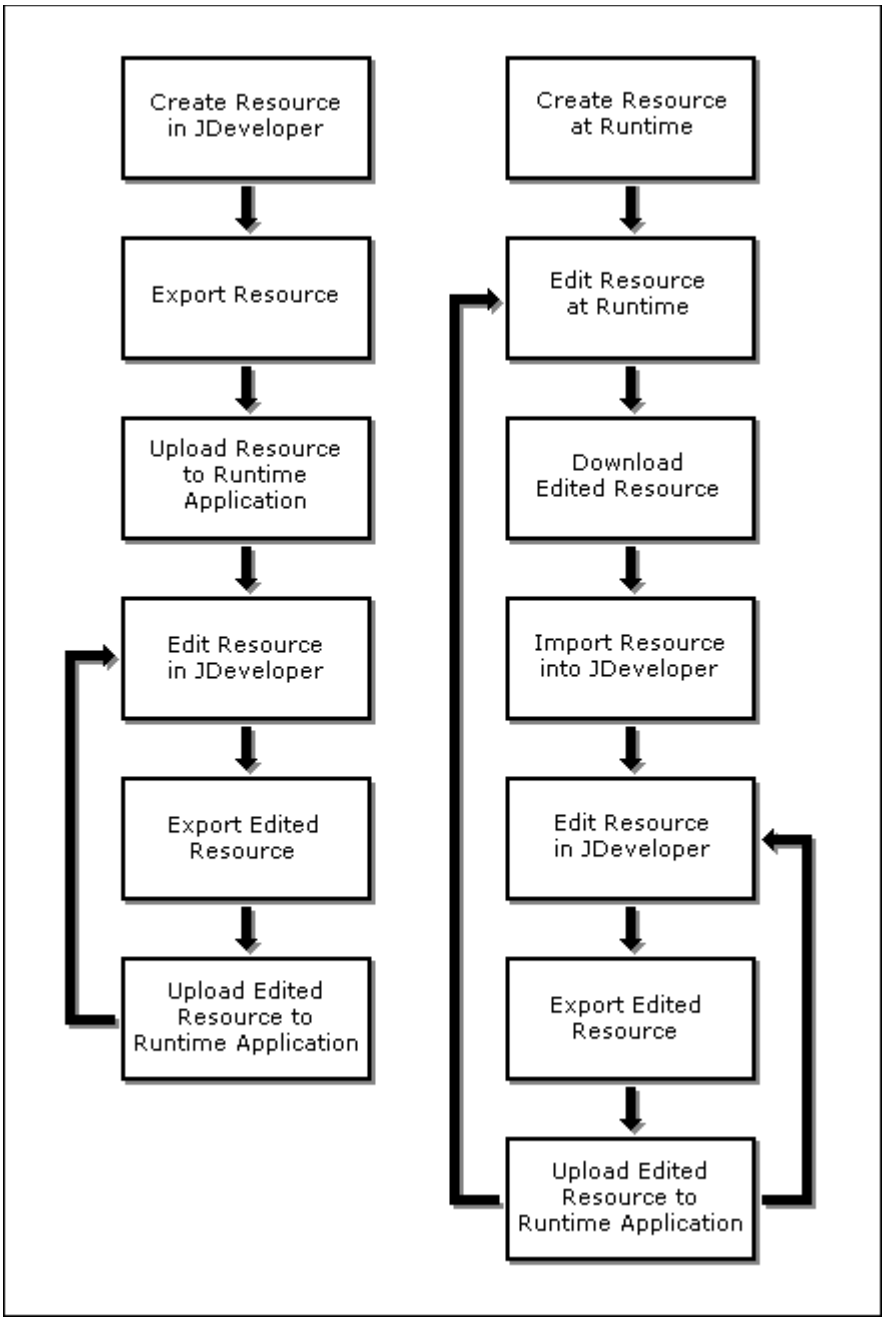
   See Section 17.4.3, "How to Export a Portal Resource from JDeveloper"
8. Upload resource to deployed application using the Resource Manager

   See Section 17.4.4, "How to Upload a Resource Using the Resource Manager"
9. Repeat from step 3.

In addition, after the deployment of the application, developers can continue to create and edit resources in JDeveloper and upload those resources to the deployed application periodically.

Figure 17–4 illustrates the round-trip development process described above.

Figure 17–4 Round-Trip Development



This section includes the following subsections:

- Section 17.4.1, "How to Download a Resource Using the Resource Manager"
- Section 17.4.2, "How to Import a Portal Resource into JDeveloper"
- Section 17.4.3, "How to Export a Portal Resource from JDeveloper"
- Section 17.4.4, "How to Upload a Resource Using the Resource Manager"

17.4.1 How to Download a Resource Using the Resource Manager

To edit a resource in JDeveloper, you first need to use the Resource Manager in the deployed application to create an EAR file that contains all the metadata for the resource.

To download a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to download.
3. In the right panel, select the resource that you want to download.
4. From the menu bar, click **Download**.
5. In the Download dialog (Figure 17–5), in the **Archive File Name** field, enter a name for the export archive (EAR) file.
6. Specify where you want to save the export archive file:
   - **Save to my computer**—Click to save the export archive file to your local file system.
   - **Save to WebCenter Portal server**—Click to save the export archive file to a remote server file system. In the **Path** field, enter the server path.

Figure 17–5 The Download Resource Dialog



7. Click **Download**.

   If you selected **Save to my computer**, you may be prompted to save the file and select the location for the saved file.

17.4.2 How to Import a Portal Resource into JDeveloper

When you have the EAR file, you must then import it into JDeveloper, where you can edit it.

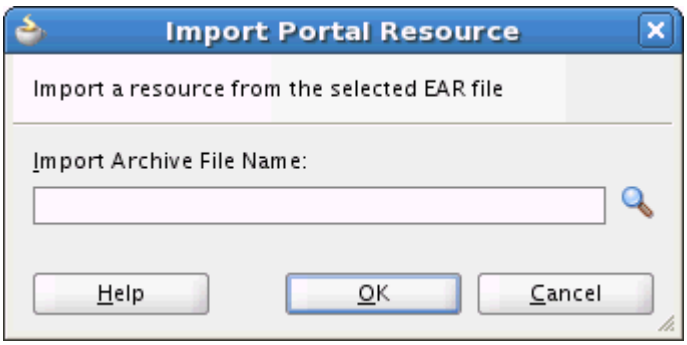
You must import the resource into an application that uses WebCenter Portal's Framework application template or that includes the appropriate technology scopes.

Note: When you import a portal resource into JDeveloper, if the resource already exists, the resource in JDeveloper is overwritten with the one from the archive. The original resource is saved to a temporary location, which is identified in the log.

To import a portal resource into JDeveloper:

1. In the Application Navigator, right-click the **Portal** project of the application into which you want to import the resource, and choose **Import Portal Resource**.
2. In the Import Portal Resource dialog (Figure 17–6), in the **Import Archive File Name** field, enter or browse for the archive file that contains the resource that you want to import.

Figure 17–6 The Import Portal Resource Dialog



3. Click **OK**.
4. If the selected archive file contains a content directory, a popup dialog requests whether you want to overwrite the existing target content directory with the content directory from the archive file. Click **Yes** to overwrite the target content directory with the content directory from the archive file. Click **No** to update the target content directory with only new files from the archive content directory, that is files that do not already exist in the target directory.

   The imported resource is created in:

   Application_Root`/oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed`

   The resource is automatically created as a portal resource.

17.4.3 How to Export a Portal Resource from JDeveloper

After editing the resource, you must export it to create an EAR file that can be uploaded to the deployed application.

Note: If you edit or add dependent objects to an imported runtime-created resource, those objects must be placed at the following path, which is the default content directory for resources created at runtime:

```
/oracle/webcenter/siteresources/scopedMD/shared
```

Dependent objects must always be placed directly under or in a subdirectory of the directory set as the content directory of a portal resource. You can find out the content directory path from the Update Portal Resource dialog.

To prevent any conflicts, you may want to place dependent objects in a subdirectory under the shared directory. For example, images for a page template named `Template1` could be stored in a `pageTemplates` subdirectory. In such a case, you must update the content directory path to point to the new location.

You can reference dependent objects in a portal resource as shown in the following example:

```
<af:image
source="/oracle/webcenter/siteresources/scopedMD/shared/pageTemplat
es/Template1/logo1.gif" id="pt_12">
```

To export a a portal resource from JDeveloper:

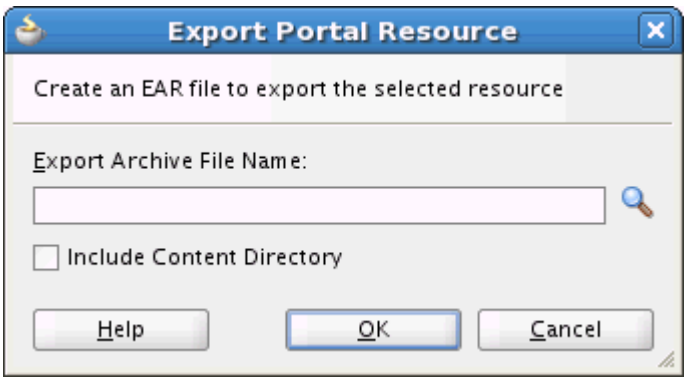
1. In the Application Navigator, right-click the resource that you want to export and choose **Export Portal Resource**.

   Note: If you do not see the **Export Portal Resource** option in the context menu, the resource is not a portal resource. For information about how to create the resource as a portal resource, see Section 17.2.1, "How to Add a Resource to the Resource Manager."

2. In the Export Portal Resource dialog (Figure 17–7), in the **Export Archive File Name** field, enter or browse for the path and file name for the export file.

   If you select an existing file, the contents of the file will be overwritten.

Figure 17–7 The Export Portal Resource Dialog



3. Select **Include Content Directory** to include the directory identified as the resource's content directory in the export archive file.

Note: This checkbox appears only for portal resources that point to a valid content directory.

The entire resource content directory is included in the export archive file; you cannot select individual files. On import, if a content directory is detected in the archive file, the user can specify whether to overwrite all files in the target content directory, or just add any new ones.

4. Click **OK**.

   The export archive contains the files that make up the resource, for example, the `.jspx` file, the page definition, and (optionally) the content directory pointed to by the resource, if it is not empty. The archive also contains the `generic-site-resources.xml` file and the registry file with just the information about the exported resource.

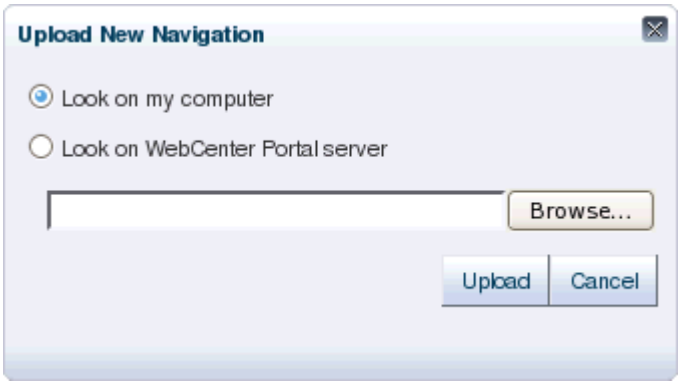
17.4.4 How to Upload a Resource Using the Resource Manager

The final step is to take the EAR file of the updated resource and use the Resource Manager to upload the resource back into the deployed application.

To upload a resource:

1. In your Framework application, navigate to the Resource Manager. For more information, see Section 17.3.2.1, "Accessing the Resource Manager."
2. In the left panel, select the type of resource that you want to upload.
3. From the menu bar, click **Upload**.
4. In the Upload dialog (Figure 17–8), specify the location of the resource archive file. Depending on the location of your EAR file, select:
   - **Look on my computer**—Select if the archive is located on your local file system. Enter the path to the EAR file or use the **Browse** button.
   - **Look on WebCenter Portal server**—Select if the archive is located on a remote server. Enter the path to the EAR file.

Figure 17–8 The Upload Resource Dialog



5. Click **Upload**.
6. Click **Yes** in the Confirm Replace dialog.

7. If the export archive file includes a content directory, the Overwrite Content Directory dialog displays. Click **Yes** to import all files from the content directory of the export archive or click **No** to import only new files from the content directory.

Part III

Customizing Your Application and Extending Customization Options

Part III contains the following chapters:

- Chapter 18, "Introduction to Composer"
- Chapter 19, "Enabling Runtime Creation and Management of Pages"
- Chapter 20, "Enabling Runtime Editing of Pages Using Composer"
- Chapter 21, "Extending Runtime Editing Capabilities Using Composer"
- Chapter 22, "Performing Advanced Composer Configurations"
- Chapter 23, "Performing Composer-Specific MDS Configurations"
- Chapter 24, "Modifying Default Security Behavior of Composer Components"
- Chapter 25, "Customizing Task Flows"

18

Introduction to Composer

Oracle WebCenter Portal's Composer is a fully-integrated page editor for adding and editing page content at application runtime. This chapter introduces Composer and describes the various tasks it supports at runtime.

To understand the functions of Composer, let us first look at a typical application development lifecycle, which comprises the following stages:

- Application developers create applications at *design time*.

  Design time traditionally represents an IDE-based environment for creating or editing applications. For WebCenter Portal, Oracle JDeveloper provides the design-time environment.

- Application administrators deploy these applications to managed servers.
- End users access the deployed applications at *runtime*.

  Runtime represents a browser-based environment used for accessing web applications.

You can see that each stage involves different categories of users. Many times, some or all of these users may want to modify pages at runtime. For example, consider a dashboard-like application that displays information pertaining to the logged-in user, for example a worklist and a mail task flow. The page developer creates the dashboard page and populates it with the required components and task flows. An application administrator deploys the application to a customer site. At the customer site, people at different levels in the organization view the dashboard and make the following requests:

- Department administrators want to add a "Department News" task flow.
- LOB administrators want to add an "Expense Incurred by Self/Directs" task flow.
- The country site administrator wants to add a task flow displaying the respective country's operations or rules.

In addition, content contributors, moderators, and end users may want to add, delete, or modify page artifacts based on the business needs. In a typical application development environment, these requests are passed on to the page developer. The developer modifies the application in the development environment and redeploys it to the customer site.

Rather than pushing simple modifications back to the developer, Composer, in conjunction with Metadata Services (MDS), provides an editing tool that enables business users to edit application pages at runtime. Changes made at runtime are saved as metadata, separate from the base application definitions. This concept of editing application pages at runtime is referred to as *design time at runtime*. It minimizes the need to modify the application at design time and redeploy it.

Composer provides components for controlling an application's design time at runtime behavior. This chapter describes the basic concepts and terminology you'll need to use Composer effectively. It includes the following subsections:

- Section 18.1, "Application and User Customizations"
- Section 18.2, "View and Edit Modes of a Page"
- Section 18.3, "Customizing Capabilities in Page View Mode"
- Section 18.4, "Editing Capabilities in Design View in Page Edit Mode"
- Section 18.5, "Editing Capabilities in Source View in Page Edit Mode"
- Section 18.6, "Composer Components"
- Section 18.7, "Security and Composer"

18.1 Application and User Customizations

When a user modifies a page at runtime, the changes are typically available only to that user. You can configure your application to make runtime modifications available to all or a subset of users accessing the page. Consider an example of a dashboard page displaying details about the expenses incurred by the members of an LOB. If the LOB administrator modifies the page and adds some expensing guidelines, the changes must be visible to all members. To address such requirements, Composer enables you to edit pages in different modes so that changes are saved as *user customizations* (that affect only the individual user's view of the page) or *application customizations* (that affect everyone's view of the page).

To save a user's changes as user or application customizations requires that they be saved separately at the back end. In the metadata domain, MDS allows for the saving of customizations in separate layers on top of the base application definitions. Depending on the business requirement, you can save changes in a single layer or in multiple layers.

In a single-layer configuration, changes are saved as application customizations to a common layer that is accessible to all users. In this case, you cannot allow for changes to be saved as user customizations.

In a multiple-layer configuration, changes are saved at separate locations based on specified criteria such as the mode in which the page is edited, the user editing the page, or the user role. In this type of configuration, each customization layer is insulated from the other, but the layers can be stacked over each other to provide flexibility during upgrades. For example, a user may want to change the default theme of the page being viewed. It is expected that this change be saved only for this user, without affecting others' view of the page. Therefore, you can set up things so that changes made by end users who have Personalize privilege on the page are saved as user customizations, and changes made by logged-in users with customization-level privileges are saved as application customizations. You can decide which tasks are available to users with Personalize and Customize/Edit privileges respectively.

For more information about MDS and customization layers, see Section 23.1, "Introduction to MDS."

18.2 View and Edit Modes of a Page

When a user opens an application page in a browser, the page opens in *View mode*. Additionally, Composer provides access to another mode called *Edit mode* to enable

users with appropriate privileges to edit application pages. These modes can be described as follows:

- View mode—The default view of a page when opened in a browser. In this mode, users can perform tasks such as rearranging components, collapsing and expanding components, and changing the layout. Typically, changes made to a page in View mode are available for the user alone and are saved as user customizations.
- Edit mode—The mode typically available to authenticated users with access privileges to edit page content. In addition to the tasks that can be performed in View mode, Composer provides controls for switching to Edit mode and performing tasks such as adding components, deleting components, and editing component properties. Typically, changes made to the page in Edit mode are saved as application customizations and are available for all users accessing the page.

18.3 Customizing Capabilities in Page View Mode

Composer enables users to customize pages in View and Edit modes. This section provides an overview of user customization tasks you can enable for users in page View mode. Such tasks include rearranging components, collapsing and expanding components, and changing the page layout. You can enable the capabilities described in this section by adding the following Composer design-time components to a page: `Panel Customizable`, `Show Detail Frame`, `Custom Actions`, and `Layout Customizable`. For information about these components, see Section 18.6, "Composer Components." Changes made to a page in View mode are available only to the user making these changes.

This section includes the following subsections:

- Section 18.3.1, "Rearrange Components"
- Section 18.3.2, "Change the Layout"
- Section 18.3.3, "Expand and Collapse Components"

Note: For a detailed description of the editing tasks that you can perform on a page, see "Building Pages" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

18.3.1 Rearrange Components

This capability allows users to organize page components so that the content most useful to them displays at the top of the page. Users can rearrange components in two ways:

- Dragging and dropping

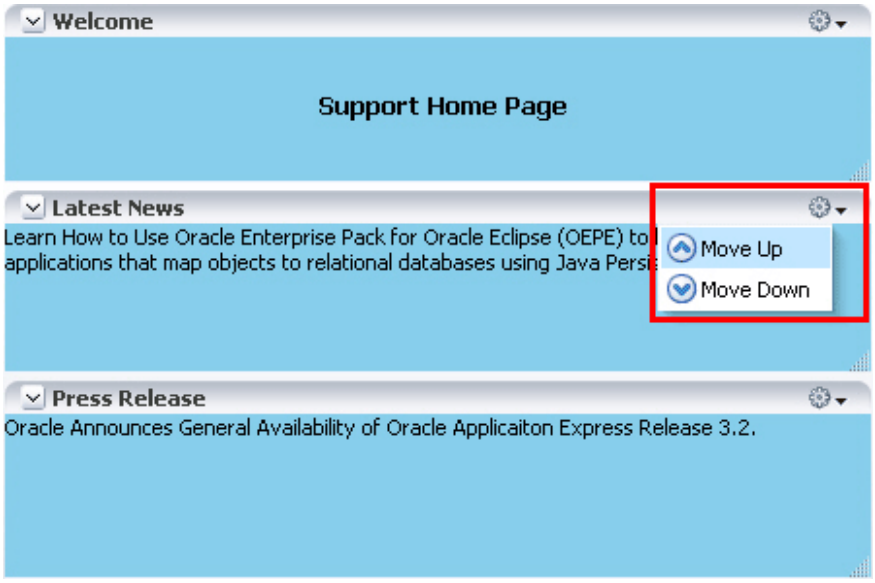
  Users can rearrange components by dragging and dropping them within the `Panel Customizable` component or across `Panel Customizable` components on the page. As it is difficult to identify a `Panel Customizable` component in View mode, users can simply drag the component over the spot they want to place it. A solid box indicates a receptive drop location.

- Using the Actions menu

  The Move actions on a `Show Detail Frame` or portlet component enable users to move these components within a parent `Panel Customizable` component. If a `Panel Customizable` has multiple child components, then a child `Show`

`Detail Frame` or portlet can be moved to the left and right or above and below, relative to the position of other child components. Figure 18–1 shows a sample `Show Detail Frame` component with Move Up and Move Down actions. In this example, selecting **Move Down** moves the Latest News component immediately below the Press Release component.

Figure 18–1 Actions Menu on a Show Detail Frame



18.3.2 Change the Layout

This capability allows users to switch between a set of predefined layouts provided by Composer. Through the use of the **Change Layout** icon, users can choose among eight different layouts to suit their needs and preferences. The `Layout Customizable` component in the Composer library enables the display of a **Change Layout** icon on the page, as shown in Figure 18–2.

Figure 18–2 Change Layout Icon

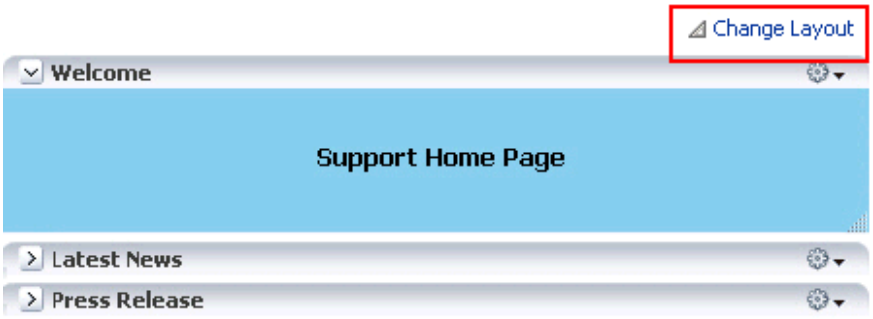
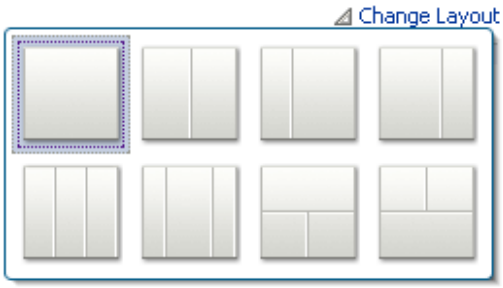


Figure 18–3 shows the layout options available with the `Layout Customizable` component. A gray colored border highlights the layout currently applied to the page or area.

Figure 18–3 Layout Options

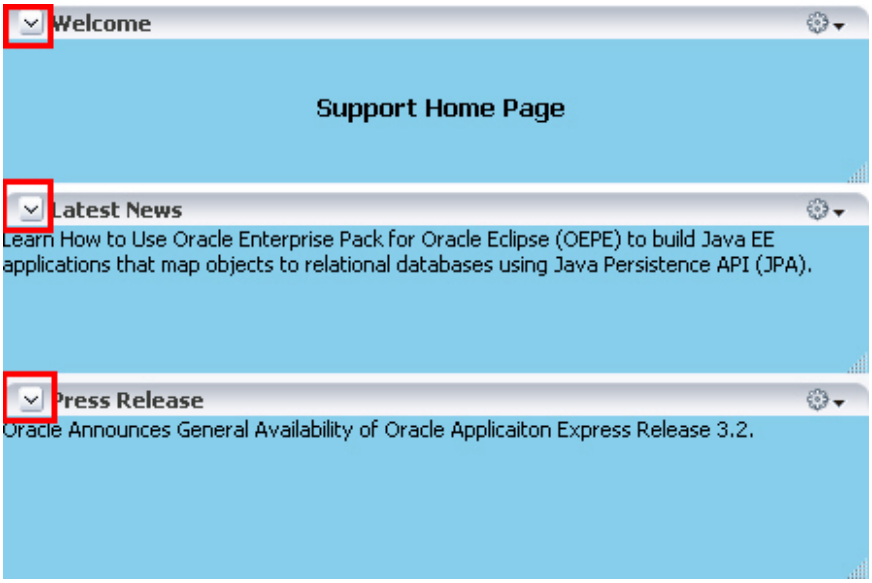


18.3.3 Expand and Collapse Components

By enabling expand and collapse components on a page, you can include more detailed information on a page without cluttering its visual appearance. Users can collapse and expand the components as desired.

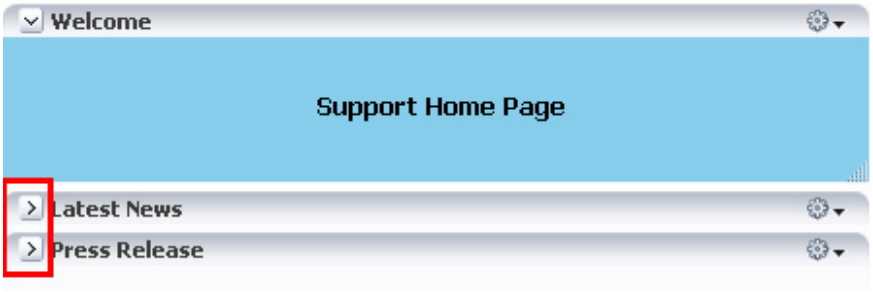
A **Collapse** icon on the header of a `Show Detail Frame` or portlet enables users to collapse such a component and display only its header. Figure 18–4 shows the **Collapse** icons on three `Show Detail Frame` components.

Figure 18–4 Collapse Icon on a Show Detail Frame



The **Expand** icon on a collapsed component enables users to disclose the component so that it displays the header and content. Figure 18–5 shows the **Expand** icons on two collapsed `Show Detail Frame` components.

Figure 18–5 Expand Icon on a Show Detail Frame



18.4 Editing Capabilities in Design View in Page Edit Mode

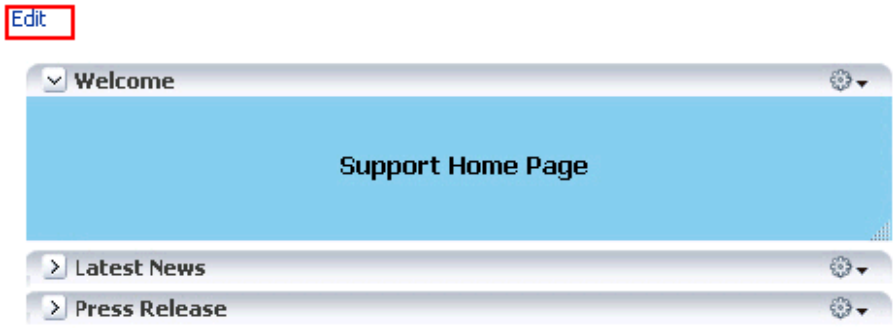
Composer's Edit mode provides a wide range of application customization capabilities. In Edit mode users can add content, edit page and component properties, delete components, rearrange components, change the page layout, and so on.

You can provide the capabilities described in this section to your application by adding the following Composer design-time components to a page: `Change Mode Link` or `Change Mode Button`, `Page Customizable`, `Panel Customizable`, `Show Detail Frame`, `Custom Actions`, and `Layout Customizable`. For more information about these components, see Section 18.6, "Composer Components."

The `Change Mode Link` or `Change Mode Button` switches the page to Edit mode. In Edit mode, users can add content to the page, edit component properties, edit page properties, and so on. The `Page Customizable` component on a page enables these editing capabilities.

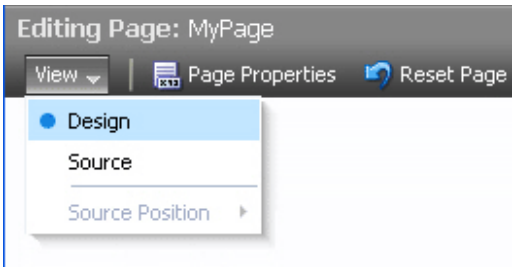
Figure 18–6 shows a `Change Mode Link` on an application page. When users click the `Change Mode Link` or `Change Mode Button`, the page opens in Composer.

Figure 18–6 Edit Link While in View Mode



The default Composer toolbar displays the name of the page being edited, a status indicator, View menu, Page Properties button, Reset Page button, and a Close icon. The View menu (Figure 18–7) provides two viewing options—**Design** and **Source**. Users can perform various editing tasks in these views.

Figure 18–7 View Menu in Composer

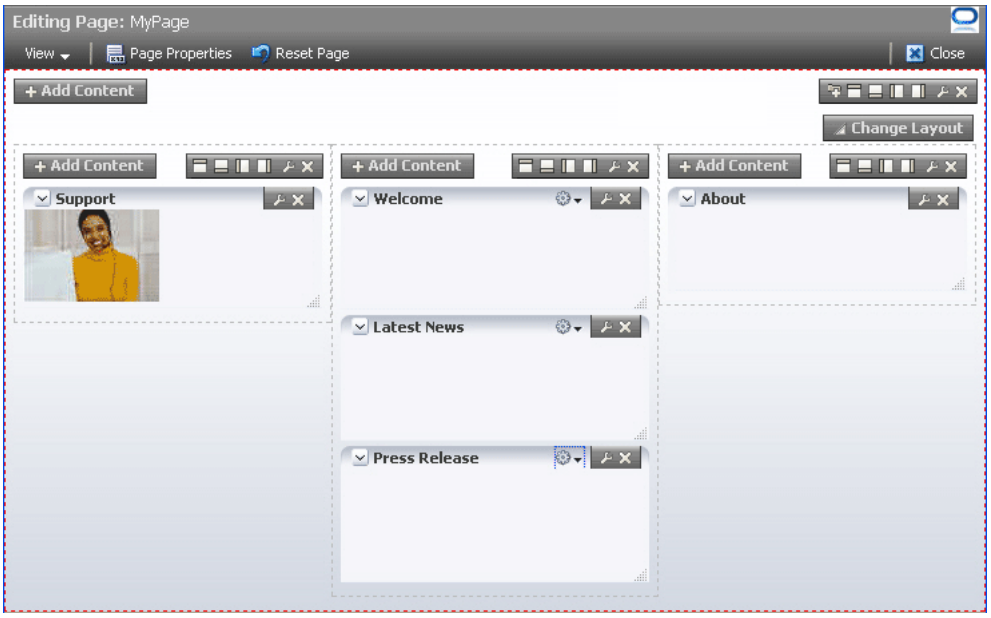


Composer

****

Design view (shown in Figure 18–8) is the default page view that provides a WYSIWYG rendering of the page and its content, where Edit and Delete controls are directly selectable on each component. In Design view users can also perform such tasks as adding content, editing page and component properties, changing page layout, and deleting components.

Figure 18–8 Design View of a Page



For information about the Source view, see Section 18.5, "Editing Capabilities in Source View in Page Edit Mode."

This section describes the editing capabilities that Composer provides in the Design view in page Edit mode. It contains the following topics:

- Section 18.4.1, "Add Content"
- Section 18.4.1.2, "Add Box Components Adjacent to Existing Ones"
- Section 18.4.2, "Rearrange Page Content"
- Section 18.4.3, "Edit Component Properties and Parameters"
- Section 18.4.4, "Reset and Override Component Properties in the Current Layer"
- Section 18.4.5, "Edit Resource Strings"

- Section 18.4.6, "Show or Hide Components"
- Section 18.4.7, "Delete Components"
- Section 18.4.8, "Change the Layout"
- Section 18.4.9, "Edit Page Properties"
- Section 18.4.10, "Wire Components to Page Parameters"
- Section 18.4.11, "Reset Page"
- Section 18.4.12, "Reset Application Customizations on Detecting Errors"
- Section 18.4.14, "Manage Application Customizations"

> **Note:** You can restrict users from performing all or some of these tasks by securing the application and applying customization restrictions on components. For more information, see Section 18.7, "Security and Composer."

Composer provides most of these capabilities in the Source view as well. The individual sections mention whether a capability is available in both modes or only in Design mode.

For a detailed description of the runtime editing tasks that users can perform on a page, see "Building Pages" and "Adding and Configuring Page Layout Components" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

18.4.1 Add Content

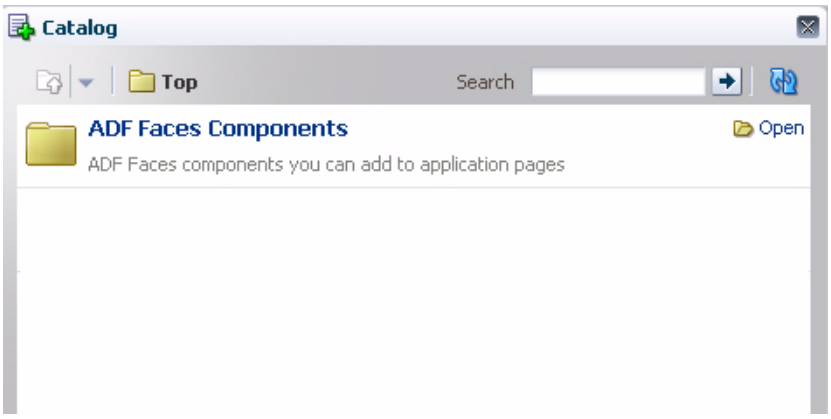
Users may want to add custom content to the page, such as a portlet displaying news, for example, or stock updates. Composer enables users to add content within any container in page Design view and Source view. WebCenter Portal's Resource Catalog, also known as *Oracle Business Dictionary*, contains resources that users can add to a page. These include documents, Oracle ADF components, portlets, and task flows. The catalog can be invoked in the following ways:

- In Design view, by clicking the **Add Content** button on any container component on the page.
- In Source view, by selecting a container component and clicking the **Add Content** button on the Source view toolbar.

> **Notes:** If you restrict customization on a container component, then users cannot add components inside it at runtime. See Section 24.1, "Applying Component-Level Restrictions by Defining Customization Policies" for more information.

Figure 18–9 shows the default catalog available in a Framework application that is not configured with portal features.

Figure 18–9 Resource Catalog

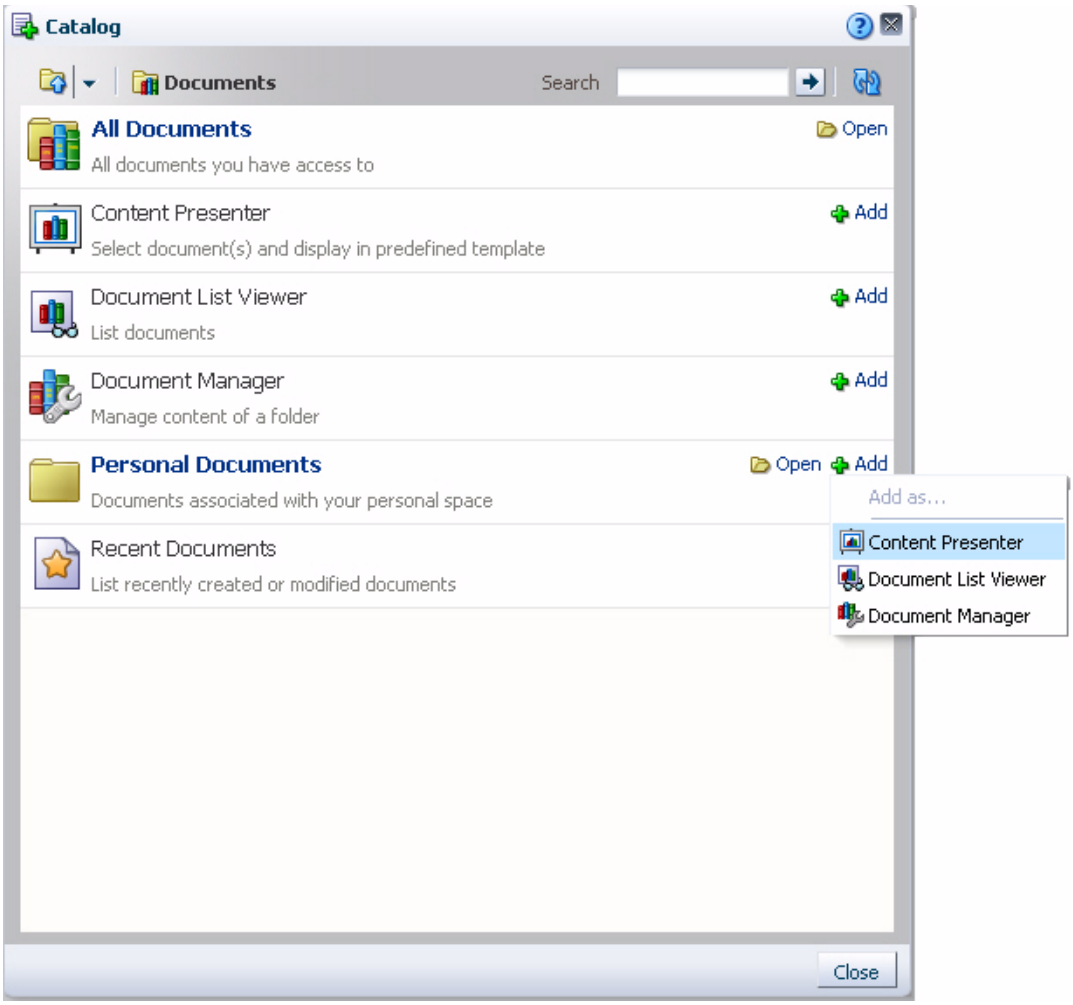


The Resource Catalog contains folders and components. An **Open** link next to an item in the catalog indicates that the item is a folder. Users can drill down into the folder by clicking this link. An **Add** link next to an item indicates that the item can be added to the page. An Add link can appear next to a component or a folder. The component is added as the first child in the container on which the Add Content link was clicked.

Note: The **Top** icon in the Resource Catalog enables users to return to a top-level folder.

If you configure drop handlers for adding resources to the page, the Add link shows a context menu with different options for adding the component to the page (Figure 18–10). For more information, see Section 21.6, "Configuring Event Handlers for Composer UI Events."

Figure 18–10 Context Menu with Drop Handlers for Adding a Component



Depending on its configuration, the Resource Catalog exposes all or a subset of the following components to users:

> **Note:** For information about Resource Catalog configurations, see Chapter 16, "Creating and Managing Resource Catalogs."

- ADF Faces Components

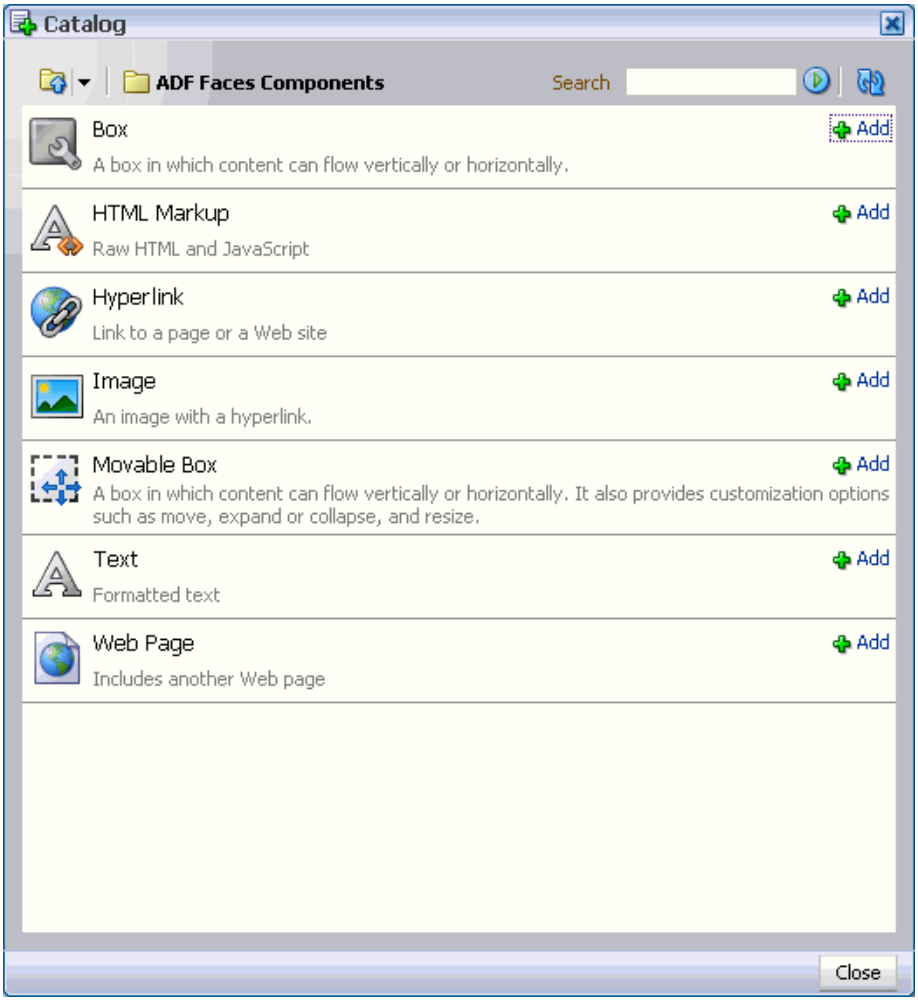
  The ADF Faces Components folder provides `Box` and `Movable Box` components that are analogous to the JDeveloper design time components `Panel Customizable` and `Show Detail Frame`. In JDeveloper, these components are available in the Composer tag library.

  > **Note:** `Panel Customizable` and `Show Detail Frame` components added to the page at design time continue to be called the same. These components are not denoted as `Box` and `Movable Box` components respectively at runtime.

This folder also provides `Image`, `HTML Markup`, `Hyperlink`, `Text`, and `Web Page` components that are analogous to the JDeveloper design time components `Go Image Link`, `Output Text`, `Go Link`, `Rich Text Editor`, and `Inline Frame` respectively. In JDeveloper, these components are available in the ADF Faces tag library.

The **ADF Faces Components** folder in the catalog contains these components, as shown in Figure 18–11.

Figure 18–11 ADF Faces Components You Can Add to the Page



- Task Flows

  Users can add custom task flows and out-of-the-box WebCenter Portal service task flows to the page. A task flow added to a page is automatically enclosed in a `Movable Box` component. As a result, the task flow displays a header with options to move or delete the component.

  Note: The Movable Box component surrounding a task flow is displayed only if the task flow is displayed on the page. That is, if you do not grant even View permission on a task flow, the Movable Box surrounding it is also not displayed on the page.

- Portlets

  The **Portlets** folder lists all the registered producers. Users can navigate to the required portlet and add it to the page. The Portlets folder exposes portlets from any Java-PDK or WSRP producer that was registered in Oracle JDeveloper. For information about registering portlet producers, see Chapter 64, "Consuming Portlets."

- Documents

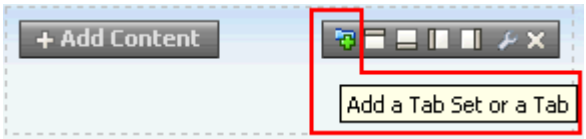
  The **Content Management** folder lists documents and folders from the Documents service that users can add to the page.

See "Adding and Configuring Page Layout Components" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* for detailed information.

18.4.1.1 Add a Tab Set or a Tab to a Box Component

The toolbar on the `Box` component provides an option (Figure 18–12) to add a tab or a set of tabs to the `Box` component. Use tabs to add content regions as separate layers on the page. Each new tab is placed as a layer below the currently selected tab and contains a region that can be populated with content.

Figure 18–12 Add Tab Set or a Tab Icon on a Box Component



When a user clicks the Add a Tab Set or a Tab icon on the Box component, Composer adds a tab around the `Box` component. In the Source view, the `Box` component is nested inside a `Show Detail Item` component, which in turn is nested inside a `Panel Tabbed` component, as shown in Figure 18–13. Users can add any number of tabs in this way.

Figure 18–13 Tab Component in Source View



The first tab that a user adds will surround the selected `Box` component. Each subsequent tab will contain a new `Box` component inside it.

Users can delete a tab by clicking the Delete icon on a tab in Design view. On deleting the last tab, the `Panel Tabbed` component is deleted from the page. Alternatively, the tab set can be deleted by deleting the `Panel Tabbed` component in the Source view. To delete only the tab without deleting the `Box` inside it, users must move (cut and paste) the `Box` outside the `Panel Tabbed` component first and then delete the `Panel Tabbed` component.

18.4.1.2 Add Box Components Adjacent to Existing Ones

A toolbar on the `Box` component in Design view provides options to add another `Box` component before or after the component. This capability enables users to modify the layout of components on the page.

When a user clicks an Add icon on the `Box` component, Composer adds another `Box` component before or after it and surrounds both `Box` components with a `Panel Group Layout`. Depending on the icon used, the new `Box` component is added to the left, right, above, or below the original `Box` component, and the `layout` attribute on the parent `Panel Group Layout` component is set to `horizontal` or `vertical`. Users can add many contiguous `Box` components in this way. Every time the orientation changes for new a `Box` component, a new `Panel Group Layout` with the appropriate layout setting (horizontal or vertical) is added around the new and the existing `Box` components.

Note: The Add icons are not rendered on the component if:

- The `allowAction` attribute on the `Panel Customizable` component is set to `false`
- The `Panel Customizable` is stretched
- The `Panel Customizable` component is restricted

For more information, see Chapter 24, "Modifying Default Security Behavior of Composer Components."

Users can delete `Box` components that they added using the Add icons, in both Design and Source view.

Note: If the `Panel Group Layout` has only two `Box` components and you delete one of them, then the `Panel Group Layout` is also deleted.

The `Box` component toolbar contains the following Add Box icons in addition to the Edit icon:

- Add Box Above
- Add Box Below
- Add Box Left
- Add Box Right

18.4.1.2.1 Add Box Above Clicking the **Add Box Above** icon (Figure 18–14) adds a `Box` component above the selected `Box` component. The `layout` attribute on the `Panel Group Layout` is set to `vertical`.

Figure 18–14 Add Box Above Icon on a Box Component



Figure 18–15 shows how the Box component appears in Source view before adding another Box component adjacent to it.

Figure 18–15 Box Component Before Adding Another One Adjacent to It

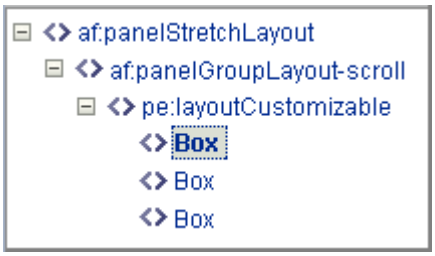
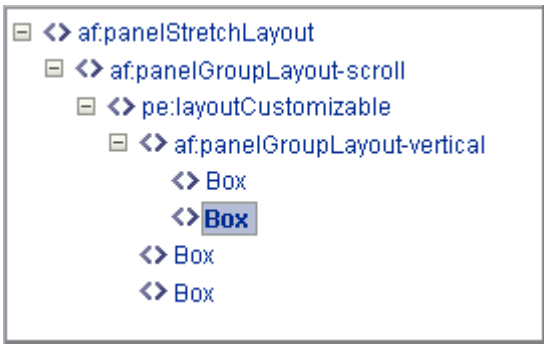


Figure 18–16 shows the two Box components vertically aligned in the component navigator.

Figure 18–16 Box Component Added Above



Users can click the **Add Box Above** icon repeatedly to add many contiguous Box components vertically within the same parent Panel Group Layout.

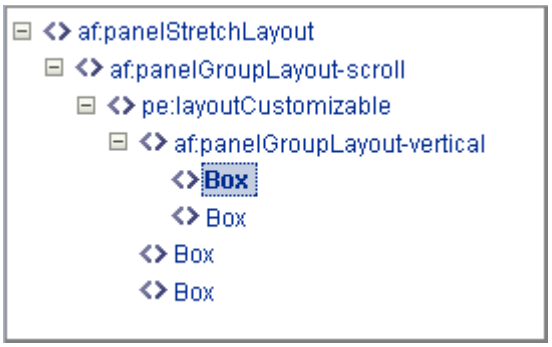
18.4.1.2.2 Add Box Below Clicking the **Add Box Below** icon (Figure 18–17) adds a Box component below the selected Box component. The layout attribute on the Panel Group Layout is set to vertical.

Figure 18–17 Add Box Below Icon on a Box Component



Figure 18–18 shows the two Box components vertically aligned in the component navigator.

Figure 18–18 Box Component Added Below



Users can click the **Add Box Below** icon repeatedly to add many contiguous `Box` components vertically within the same parent `Panel Group Layout`.

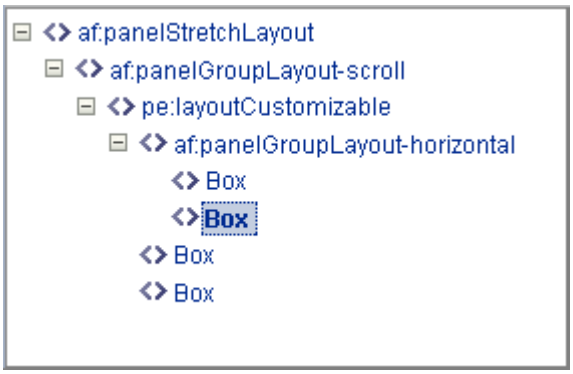
18.4.1.2.3 Add Box Left Clicking the **Add Box Left** icon (Figure 18–19) adds another `Box` component to the left of the selected `Box` component. The `layout` attribute on the `Panel Group Layout` is set to `horizontal`.

Figure 18–19 Add Box Left Icon on a Box Component



Figure 18–20 shows the two `Box` components horizontally aligned in the component navigator.

Figure 18–20 Box Component Added to the Left



Users can click the **Add Box Left** icon repeatedly to add many contiguous `Box` components horizontally within the same parent `Panel Group Layout`.

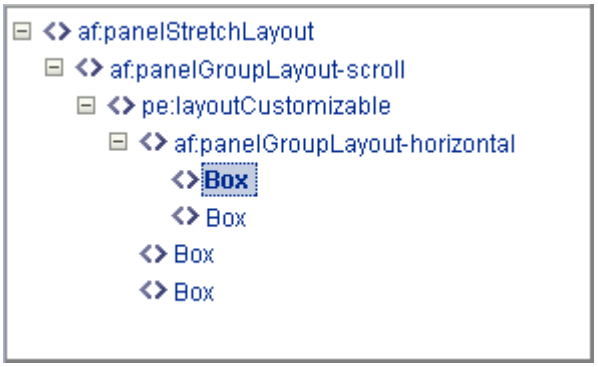
18.4.1.2.4 Add Box Right Clicking the **Add Box Right** icon (Figure 18–21) adds another `Box` component to the right of the selected `Box` component. The `layout` attribute on the `Panel Group Layout` is set to `horizontal`.

Figure 18–21 Add Box Right Icon on a Box Component



Figure 18–22 shows the two `Box` components horizontally aligned in the component navigator.

Figure 18–22 Box Component Added to the Right



Users can click the **Add Box Right** icon repeatedly to add many contiguous `Box` components horizontally within the same parent `Panel Group Layout`.

18.4.2 Rearrange Page Content

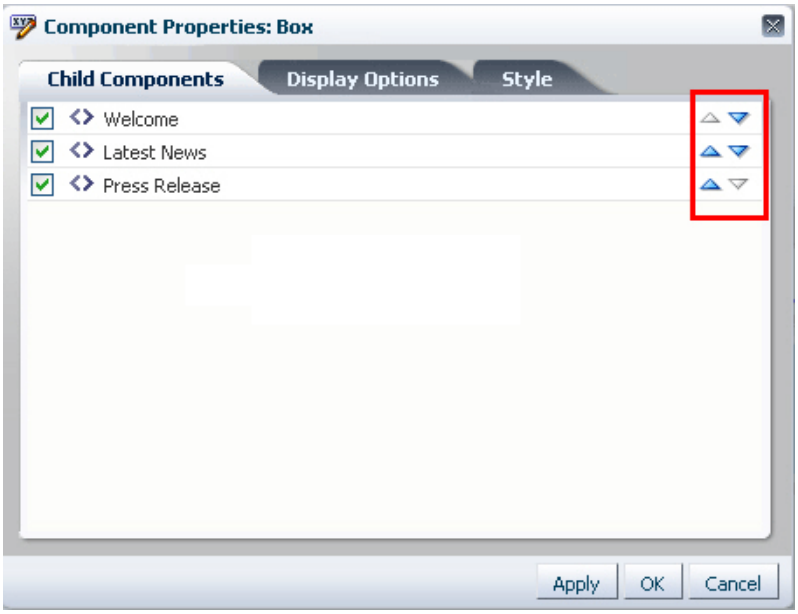
This capability allows users to organize page components based on where they are easiest to use; for example, users can move content most useful to them to the top of the page. In Design view, users can rearrange content on the page in the following ways:

- Drag and drop a component within the same `Panel Customizable` component or across `Panel Customizable` (or `Box`) components on the page.
- Use the Move actions on the Actions menu of a `Show Detail Frame` or portlet to move it within the parent `Panel Customizable` component. Depending on the number of child components in the `Panel Customizable` and how these components are oriented, a component can be moved to the left and right, or up and down.
- To resequence components within a container, in the Component Properties dialog for the container, click the Child Components tab and use the up and down arrows shown against each child component (Figure 18–23).

Notes: A component on which customization is restricted, can be rearranged inside its parent container if the parent is customizable.

The Child Components tab displays a list of direct child components at the top (Components list) of the tab, followed by a list of all facets defined for the component (Fixed Components list). There are no up and down arrows against facet components as these components cannot be rearranged. The Fixed Components list is displayed only if the component contains facets.

Figure 18–23 Resequencing Options on the Child Components Tab of a Container



18.4.3 Edit Component Properties and Parameters

The Component Properties dialog enables users to edit component properties and the parameters associated with components, such as portlets and task flows. The Component Properties dialog is accessible in both Design view and Source view.

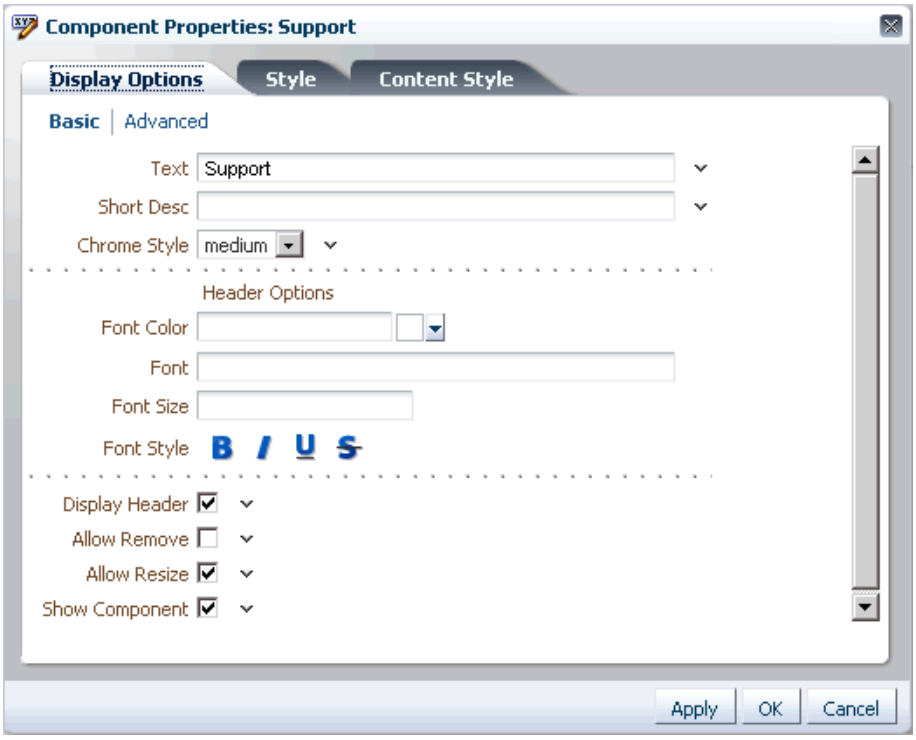
Note: A component cannot be edited if:

- The `Id` attribute was not set for the component at design time
- The component or any of its attributes have been restricted

  For information about restricting customization, see Chapter 24, "Modifying Default Security Behavior of Composer Components."
- It is a component from the ADF Faces library

In Design view, clicking the **Edit** icon on a component displays the Component Properties dialog, shown in Figure 18–24.

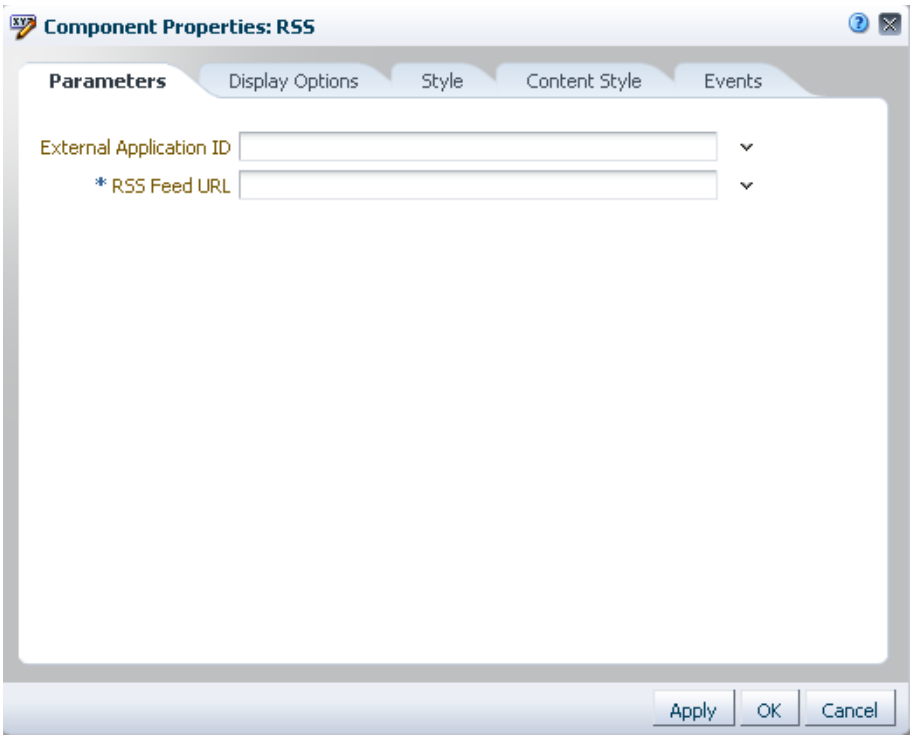
Figure 18–24 Component Properties Dialog



The Display Options tab enables users to edit visual properties, such as background and title.

The Parameters tab (Figure 18–25) enables users to edit parameters for components such as portlets and task flows.

Figure 18–25 Parameters tab in the Component Properties Dialog

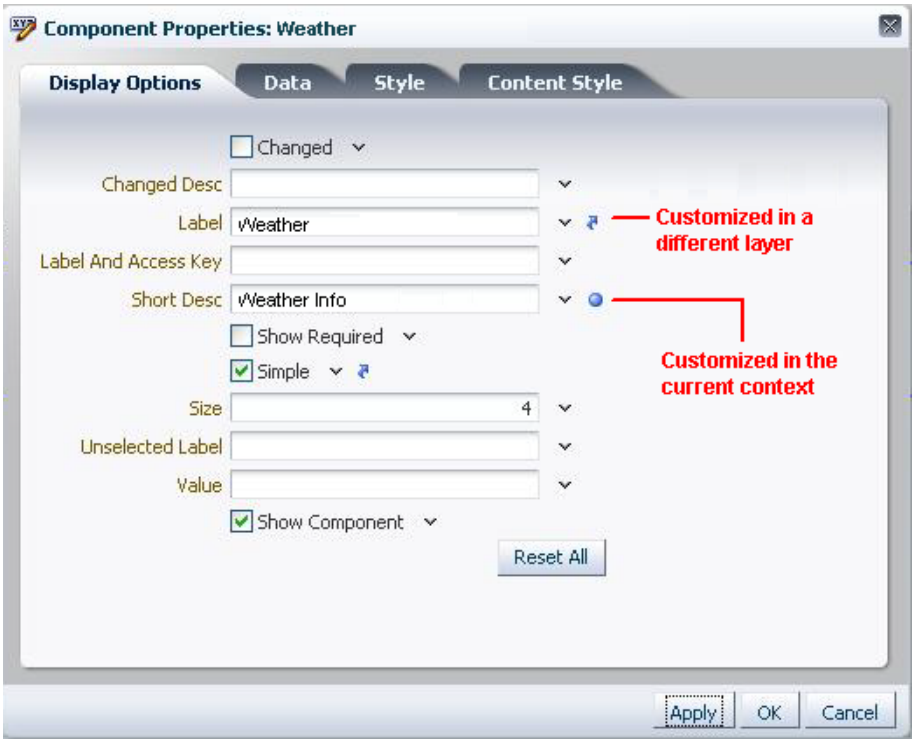


Visual Indication of Component Property Customization

An icon next to a property on the Display Options tab indicates that the property has been edited at runtime. Two different icons are used to indicate changes made in the current context and in a different layer. Figure 18–26 shows the two different icons used to indicate that the property was updated.

Note: Visual indication of property customization is relevant only if you have configured multiple customization layers in your application. For information, see Section 23.3, "Adding Customization Layers to View and Edit Modes: Example."

Figure 18–26 Visual Indication of Property Customization



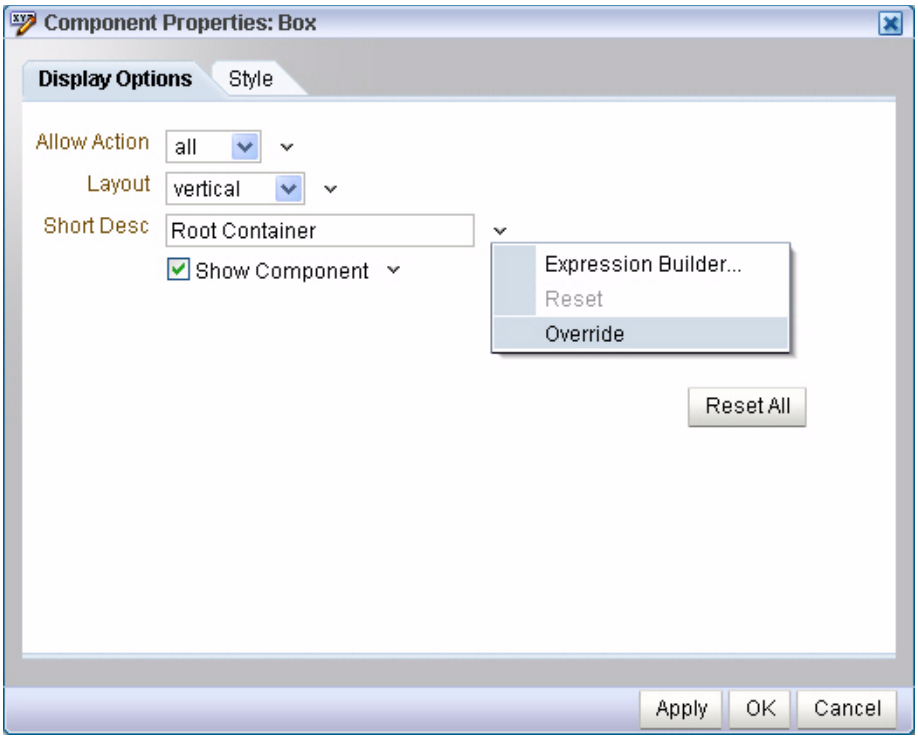
The Component Properties dialog also enables users to reset visual properties in the current context. For more information, see Section 18.4.4, "Reset and Override Component Properties in the Current Layer."

18.4.4 Reset and Override Component Properties in the Current Layer

Users can reset and override application customizations made to a component's properties in the current layer. The Component Properties dialog provides options to reset a particular property or all properties on that panel in the Component Properties dialog.

An **Override** option on the context menu for a property (Figure 18–27) enables users to use a value from another layer as a fixed value for the current layer. For example, a user edits the `Short Desc` property on a `Movable Box` component in layer `X` and sets it to `First Frame`. A user editing this component in another layer, `Y`, can choose `First Frame` as the fixed value for `Short Desc` in layer `Y`. All users that are accessing the component in layer `Y` will see this value henceforth. The value for `Short Desc` does not change in layer `Y` even if it is changed in layer `X`. When a user enters a value for a property, the Override option for that property is grayed out.

Figure 18–27 Reset and Override Options in the Component Properties Dialog



A **Reset** option on the context menu for a property enables users to reset that property to its original state. The Reset option is grayed out when the property has not yet been edited in that layer, or when the user has chosen to override the property with the displayed value, which comes from another layer.

A **Reset All** button enables users to reset all properties on the Display Options tab. Users can reset only application customizations applied to visual properties, which are rendered on the Display Options tab.

Note: The Reset option and the Reset All button are not available if the component was created newly or none of the attributes were modified in the current layer.

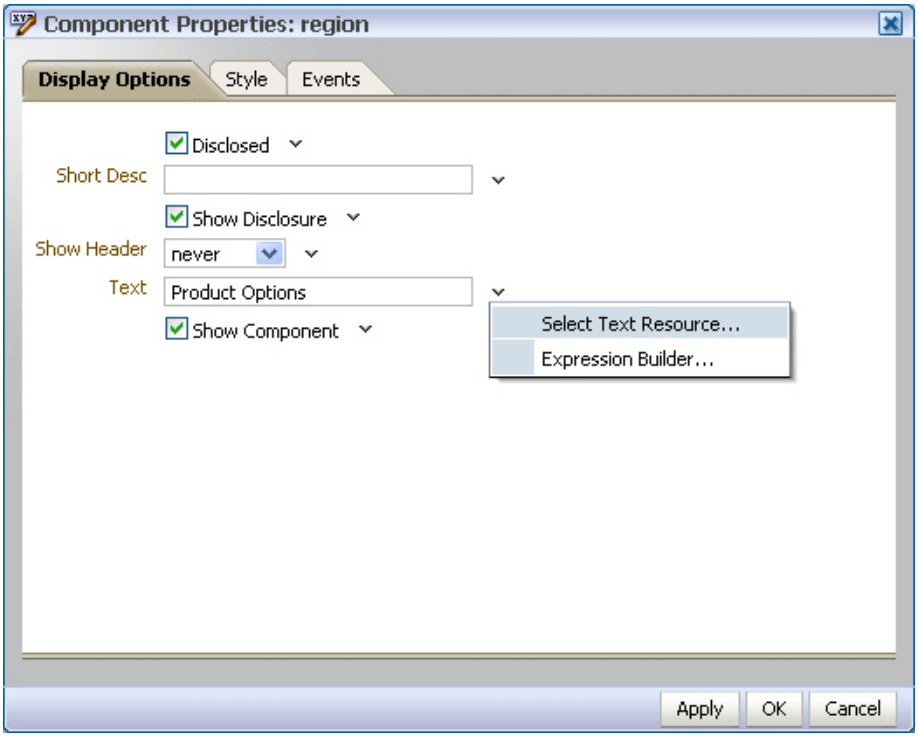
18.4.5 Edit Resource Strings

Component properties edited in Composer are available only in the current language. For example, if you add a task flow to your page, the task flow title you enter in Composer is available only in the current language. As a result, users in different locales do not see the translated values for such properties. To provide language support for component properties edited at runtime, Composer enables users to edit resource strings for properties that take String values. This is similar to the resource string editor feature provided by JDeveloper. Changes made to resource strings in Composer are saved into an application override bundle. This bundle is sent for translation and the translated versions are imported back into the application. This way, users are able to see the property values in their language.

For more information about the override bundle and configuring resource string editing in existing (release 11.1.1) Framework applications, see Section 22.12, "Configuring Runtime Resource String Editing."

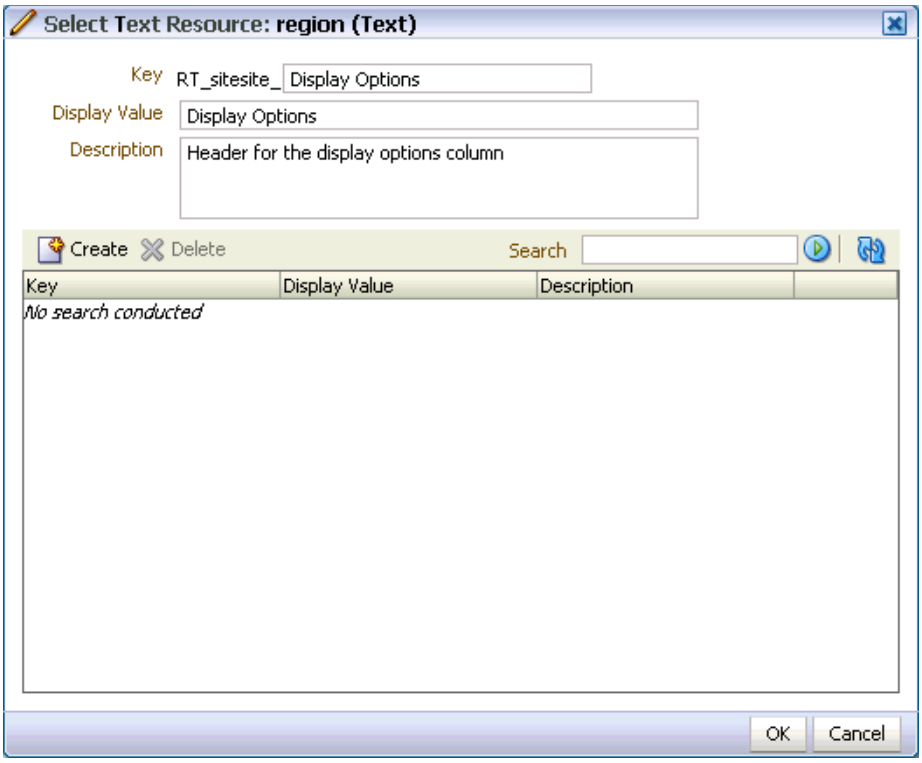
At runtime, for component display options that can take String values, the Edit menu next to the property field provides a Select Text Resource option, as shown in Figure 18–28. Users can select this option to edit resource strings for properties.

Figure 18–28 Select Text Resource Option on an Attribute



The Select Text Resource dialog (Figure 18–29) provides options to search existing resources, edit or delete resource keys, and add new resource key/value pairs.

Figure 18–29 Select Text Resource Dialog



The Select Text Resource dialog displays the following:

- Key field: For specifying a new key name or modifying the existing name. This is a String value used for uniquely identifying a locale-specific object in the resource bundle.

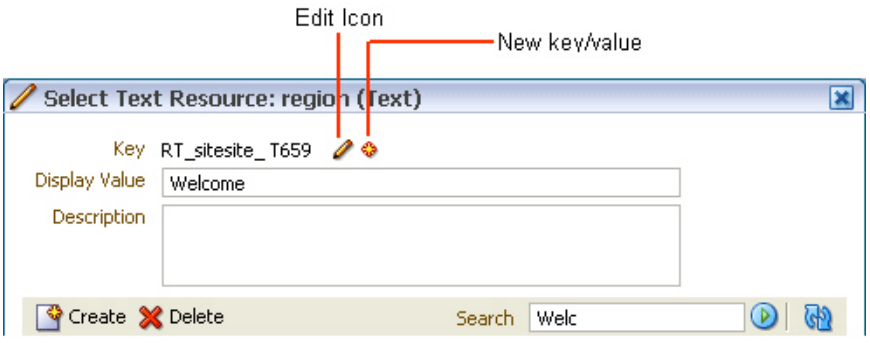
  The key name is automatically prefixed with the MDS layer to which the resource belongs. The prefix helps distinguish keys created at runtime from those created at design time.

  The **Edit** and **New Key/Value** icons (Figure 18–30), displayed for existing keys, enable users to perform the following tasks:

  – **Edit**: Modify the key value for an existing resource string. The string with new values is saved into the override bundle.

  – **New Value/Key**: Create key and value pairs in the application override bundle.

Figure 18–30 Icons for Creating and Editing Resource String Keys

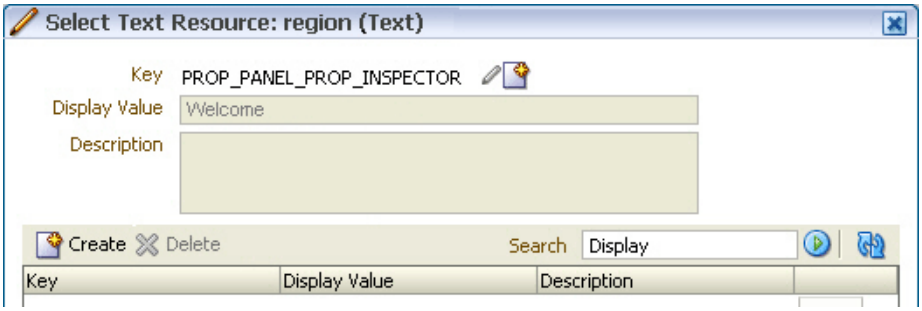


Users can edit only strings that were created in the same MDS layer. Other strings appear as read-only, as shown in Figure 18–31.

Note: When you try to override the default content in the resource bundle by directly entering values in the Select Text Resource dialog, the changes do not take effect and the page may appear blank.

Therefore, it is recommended that you create a new resource string instead of directly entering values in the Select Text Resource dialog.

Figure 18–31 Read-Only Resource Strings in the Select Text Resource Dialog



- Display Value field: For specifying a resource string to translate for display in the UI or modifying an existing string.
- Description field: For specifying a translatable description of the resource or modifying an existing description.
- Search field: For searching existing resources. Users can search for strings that were created in the same layer, in a different MDS layer, or resource bundles defined and used in a JSPX file at design time.
- Search results table: For displaying search results, editing strings, and specifying values for new strings.

  Users can edit strings that were created in the same layer. Other strings created at design time or in a different MDS layer can only be used for the property, but not edited.

  A link or a status indicator toggle displays in the last column and enables users to promote a string to be the active, to-be-translated resource (Figure 18–32) or identifies a string as the active resource (Figure 18–33).

Figure 18–32 Use Link on a Resource String

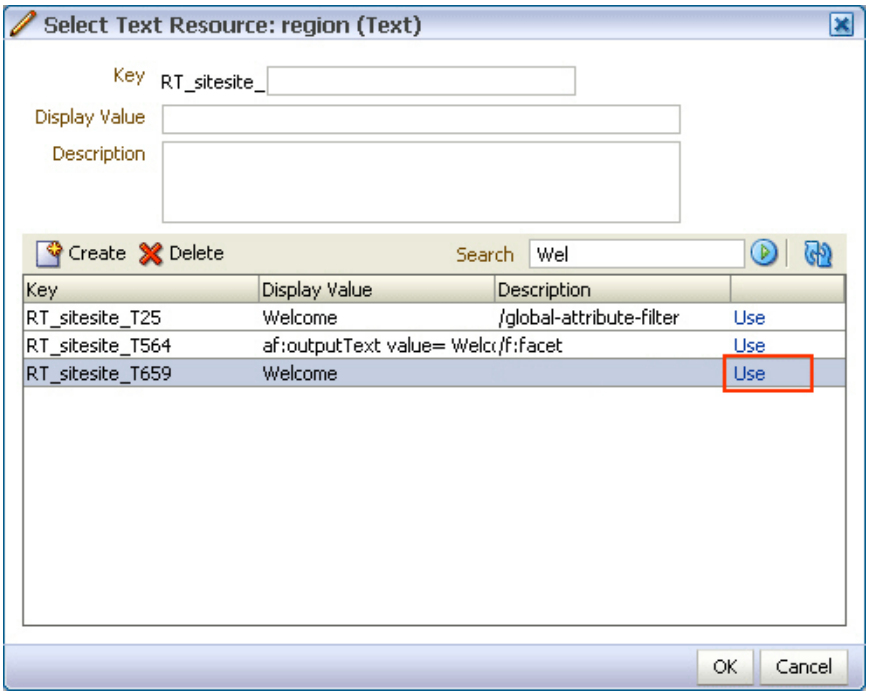
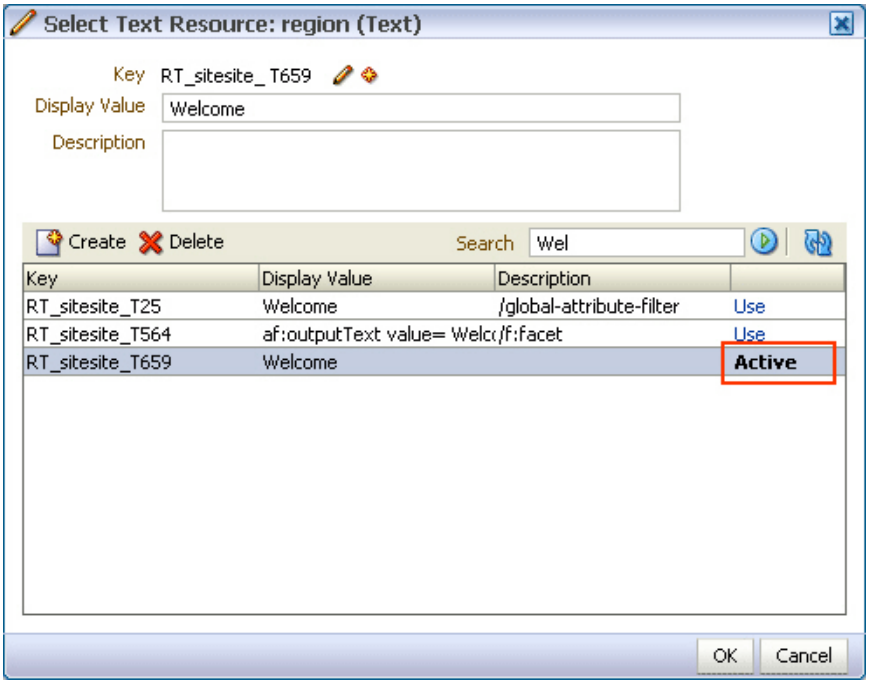


Figure 18–33 Active Status on a Resource String



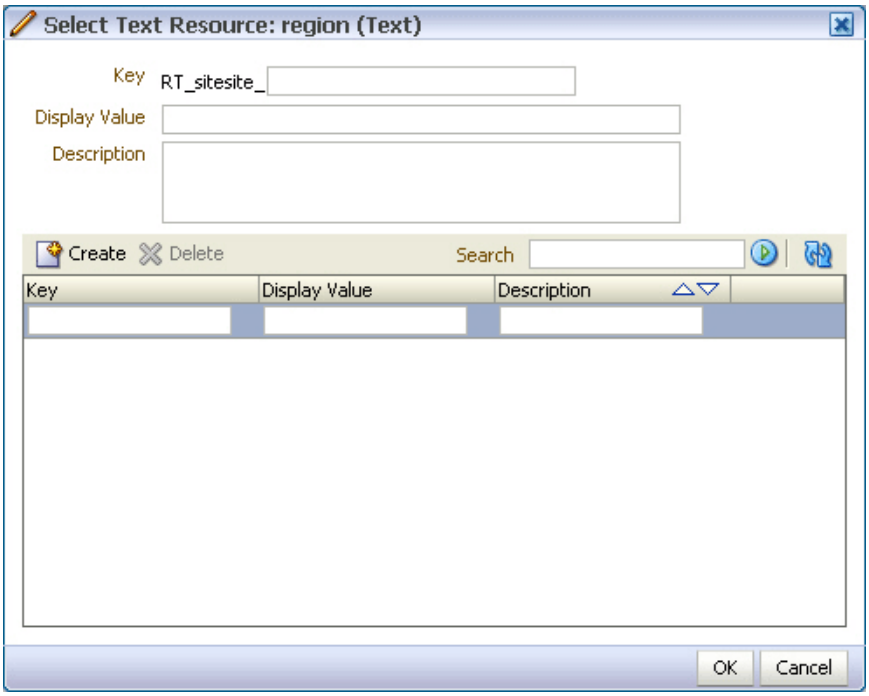
Note: When searching for resource strings created at design time, Composer searches for the `c:set` tag that is used when specifying resource strings in JSPX files.

The following example shows the `c:set` element used for defining the `ComposerBundle` in a JSPX page:

```
<c:set var="portalBundle"
        value="#{compBundle['test.resource.ComposerBundle']}"/>
```

- Create button: For creating a new resource string. On clicking the **Create** button, the search results table provides a new row with input fields for creating a new resource. Users can double click inside the row and create a new string by specifying values for the Key, Display Value, and Description, as shown in Figure 18–34, and then click **OK**.

Figure 18–34 Fields for Creating a New Resource String in the Search Results Table



Users can just create inactive strings that get saved to the override bundle. Clicking the **Use** button in the last column selects this string for use as the current property value.

- Delete button: For deleting a selected resource from the search results table.
- Refresh: For refreshing the search results table.

Notes

- Composer supports creation of around 500 resource strings at runtime. Beyond this number, application performance slows down.
- The search criteria in the resource string editor is case-sensitive.

- To search for the description of a resource string, the user must provide the complete string value in the search criteria.
- The resource string editor does not display string descriptions for entries created in Properties bundle or List Resource bundle formats in JDeveloper.
- The option to edit resource strings is disabled for properties on the Parameters and Events tabs.

  If the EL value for an ADF resource is referenced in a page definition parameter such as page parameter, task flow parameter, or portlet parameter, then the binding EL returns an empty value. Users must avoid referencing ADF resource EL values in page definition parameters.

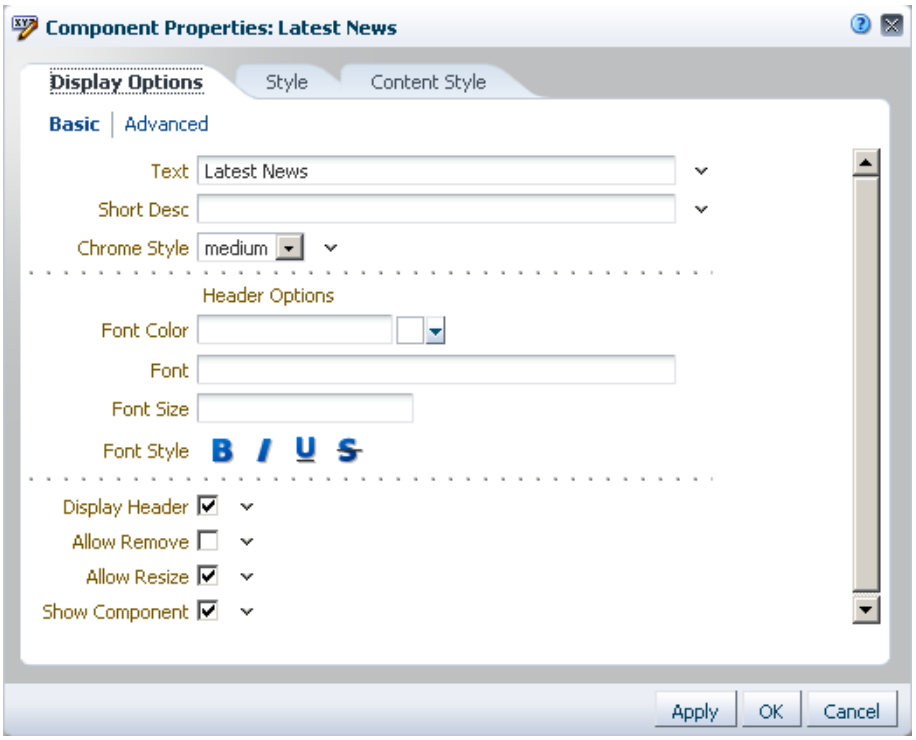
18.4.6 Show or Hide Components

Users can choose to show and hide components selectively on the page using various options. Users can hide or show a component in the following ways:

Note: Show or hide behavior is tied to the component's `rendered` property. When a user hides a component, its `rendered` property is set to `false` and vice versa. If the `rendered` attribute on a component has an EL value, then that value is lost on using the show or hide option. The value is set to `true` or `false`.

- Click the **Edit** icon on the component. In the Component Properties dialog, clear the Show Component check box (Figure 18–35).

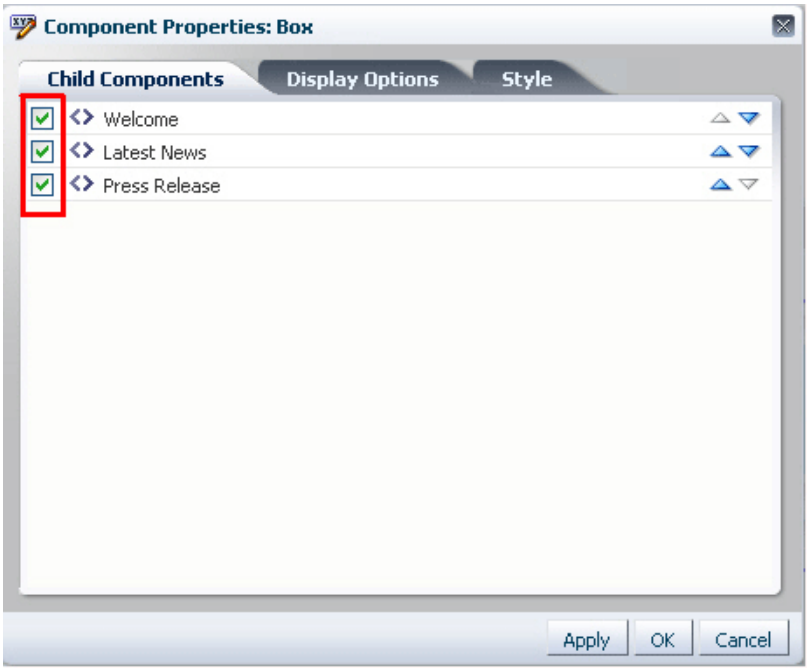
Figure 18–35 Show Component Option in the Component Properties Dialog



Checking the Show Component check box and clicking Apply or OK in the dialog renders the component on the page again.

- Click the **Edit** icon on the component's container. On the Child Components tab in the Component Properties dialog, click the Hide Component link shown against the component name.

Figure 18–36 Hide Component Link in a Container's Component Properties Dialog



When a component is hidden, the Component Properties dialog for its container displays a Show Component link against that component. Clicking this link and then clicking Apply or OK in the dialog displays the component again.

18.4.7 Delete Components

Users can delete a component from the page by clicking the **Delete** icon on its header.

A Delete dialog prompts users to confirm deletion.

Note: If you defined customization restrictions on a component, then the **Delete** icon is disabled for the component at runtime. For more information, see Section 24.1.1, "How to Define Type-Level Customization Policies."

18.4.8 Change the Layout

In Design view, the **Change Layout** icon enables users to select a layout from a set of eight predefined layouts. The default page layout is `threeColumn`.

In Source view, users can select the `Layout Customizable` component and view its properties. The layout options are displayed in the Component Properties dialog. Users can select any predefined layout and click Apply or OK.

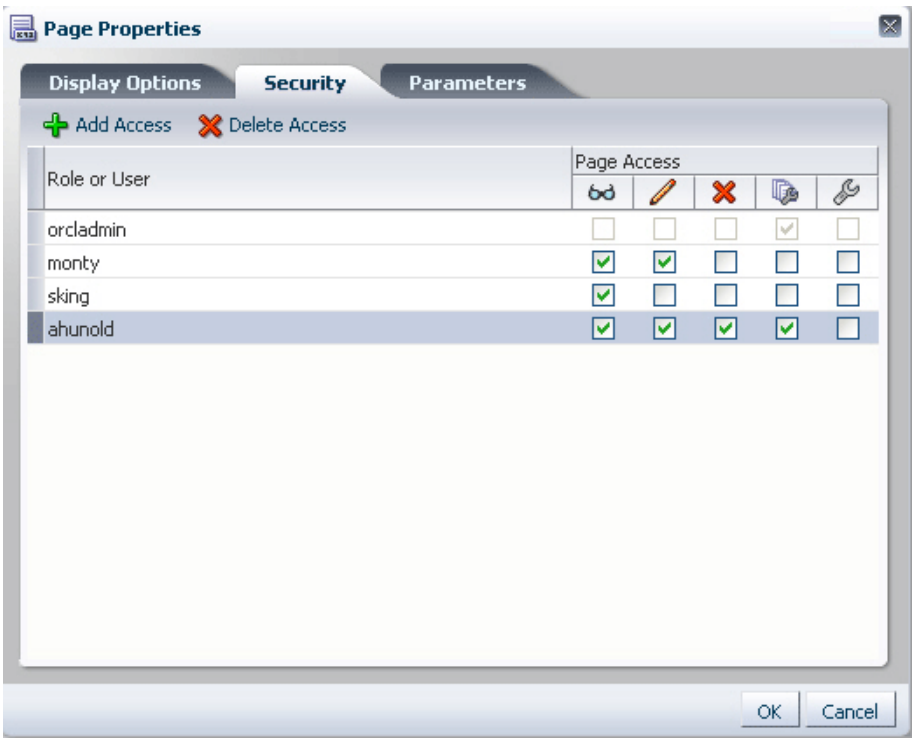
Note: Predefined layout options are available to users only if you have added a `Layout Customizable` component to the page at design time. For more information, see Section 18.6.5, "Layout Customizable."

18.4.9 Edit Page Properties

Users can access the **Page Properties** dialog from both Design view and Source view. In this dialog, users can create or edit a page's parameters and display properties. For pages created at runtime in a secured application, a Security tab enables users with Grant privilege to edit page access privileges for other users.

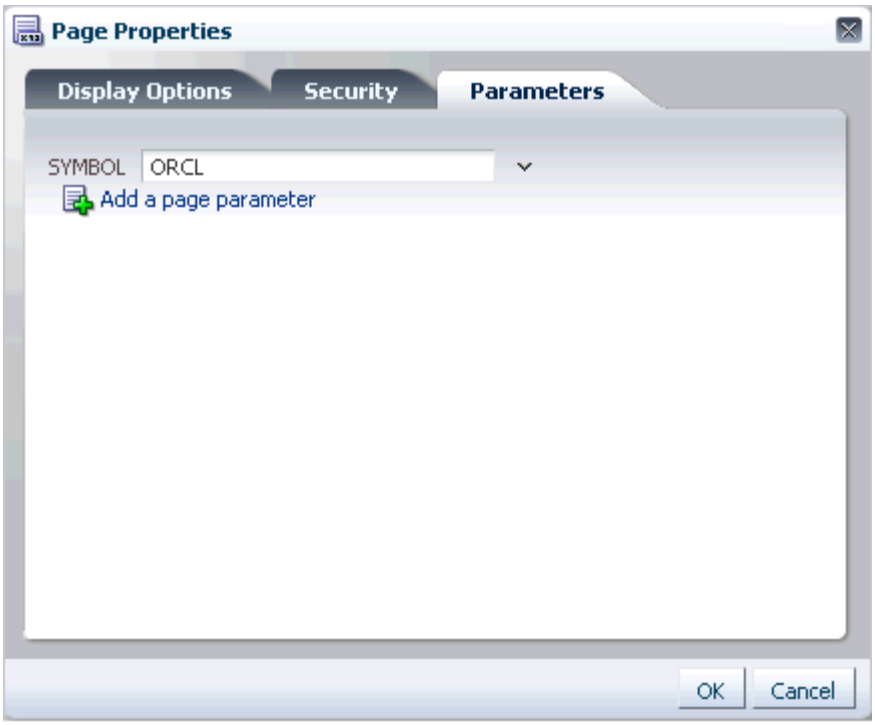
Figure 18–37 shows the Security tab in the Page Properties dialog.

Figure 18–37 Security Tab on the Page Properties Dialog



On the Parameters tab (Figure 18–38), users can create page parameters that can be linked to component properties, thereby enabling any component on the page to adapt to the page context. For example, suppose that a page has a parameter called `SYMBOL` with the default value `ORCL`. This page contains the Stock Chart, Stock Price, and Company Info task flows. The task flows can be linked with the page parameter so that all the task flows respond to a change in the value of `SYMBOL` and are updated accordingly.

Figure 18–38 Parameters Tab on the Page Properties Dialog



See "Building Pages" and "Wiring Pages, Task Flows, Portlets, and UI Components" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* for detailed information about editing page settings.

18.4.10 Wire Components to Page Parameters

Composer enables users to link portlets and task flows with page parameters so that these components can read the page parameters and change their behavior accordingly.

Users can wire portlet and task flow parameters to page parameters using the Component Properties dialog. For more information, see "Wiring Pages, Task Flows, Portlets, and UI Components Together" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

In addition to wiring components and page parameters, Composer also enables users to pass values for display options and parameters through page URLs. For more information, see "Passing Parameter Values Through the Page URL" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

18.4.11 Reset Page

The **Reset Page** button is available on the page in both Design view and Source view. This button invokes the Reset Page dialog that provides options to remove edits made to a page in the current layer or tip layer and reset it to a previously saved version or its original, out-of-the-box state.

Note: The **Reset Page** button is a default add-on rendered on the Composer toolbar. You can disable this button if you do not want users to reset the page. For information, see Section 21.2.5, "How to Selectively Display Add-Ons."

Roll Back to a Specific Label

If your application is configured to use a database store, a new version of the page is generated each time a user saves application customizations. If a sandbox is configured for your application, by creating a label with the `SAVE_LABEL_%` convention you can ensure that Composer resets the page as follows:

- If there is only one label name following the `SAVE_LABEL_%` convention, then Composer resets the page to its state in that label.
- If there is no label name following the `SAVE_LABEL_%` convention, then Composer resets the page to its original, out-of-the-box state.
- If there are multiple labels following the `SAVE_LABEL_%` convention, then Composer resets the page to its state in the latest label among these.

To roll back to a specific label, you must have configured your application to use a sandbox. Specifically, the namespace of the page being customized must be defined in the `<metadata-namespaces>` section in the application's `adf-config.xml` file. For more information, see Section 23.2.1, "How to Enable Composer Sandbox Creation."

Rolling back to a specific label is useful if your application has dependencies on the customization metadata and resetting the page to its original state can cause issues.

Notes:

- Users can only reset application customizations done on a page. Components defined in the base page or a shared component's page are left intact.
- When a user resets a page, components added to the page at runtime are removed. However, if the page uses a template, then components added inside `Panel Customizable` components on the template are not removed.

Example 18–1 shows the sample code to create a label with a `SAVE_LABEL_` prefix.

Example 18–1 Code to Create a Label Prefixed with SAVE_LABEL_

```
import oracle.mds.versioning.VersionHelper;
import oracle.mds.naming.Namespace;

....
long labelNumber = (long)(Math.random() * 1000000000);
String savedlabel = "SAVE_LABEL_" + labelNumber;

MDSInstance mdsInstance =
(MDSInstance)ADFContext.getCurrent().getMDSInstanceAsObject();

VersionHelper vh = VersionHelper.get(mdsInstance);

Namespace[] validNamespaces = new Namespace[1];
```

```
Namespace pageNamespace = Namespace.create(<pagePath or taskflow path>,
NamespaceRestriction.CUSTOMIZATIONS);

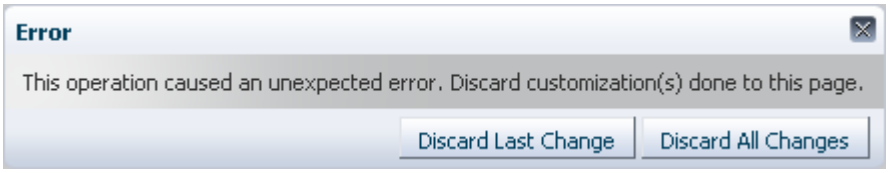
validNamespaces[0] = pageNamespace;
vh.createLabel(savedlabel, null, validNamespaces);
```

18.4.12 Reset Application Customizations on Detecting Errors

Composer provides a mechanism to alert users to customization errors that may cause a page to break. Users are alerted about errors in the following ways:

- If the change is saved to MDS, an Error dialog, shown in Figure 18–39, provides the option to roll back only the last change or all changes that users made in the current context.

Figure 18–39 Error Dialog Prompting Users to Discard Changes



- If the change is not yet saved to MDS, an Error dialog, shown in Figure 18–40, simply informs users that an error occurred and the page will be reloaded.

Figure 18–40 Error Dialog Informing Users About Page Reload



Errors occurring while performing the following operations may cause the page to break:

- Pasting a component at the wrong position while using the Cut and Paste options; for example, pasting a table column outside the table.
- Adding components or task flows; for example, adding a task flow that calls a Web service, which in turn gives an error.
- Moving `Show Detail Frame` components on the page using the drag-and-drop method.

18.4.13 Create Labels On Saving Application Customizations

A Save and Label button on the Composer toolbar, shown in Figure 18–41, enables users to save their application customizations in a new label. Labels created in Composer are stored with a prefix of `composer_`, that is, a label created with the name `myLabel` is stored as `composer_myLabel`.

Creating labels enables users to reset customizations on a selected object to those from a selected label. This can be achieved using the Promote link in Customization Manager. For more information, see the section titled Promote Customization Metadata.

Note: The Save and Label button is displayed only if you have configured a sandbox with an MDS database store for your application and set the `allowLabel` attribute on the `Page Customizable` component to `true`. For more information about this attribute, see Section B.1.1, "Page Customizable."

Figure 18–41 Save and Label Button on the Composer Toolbar



18.4.14 Manage Application Customizations

The Customization Manager is a Composer add-on panel that enables users to manage customization metadata for objects such as XML documents, pages, page fragments, and task flows on the page. It displays a list of all such objects on the page, and provides details about the layers in which these objects are customized. Users can download customization metadata for a selected object, edit it, and upload the revised metadata file. Additionally, the Customization Manager also provides options to delete customizations and promote customizations from a previously saved label.

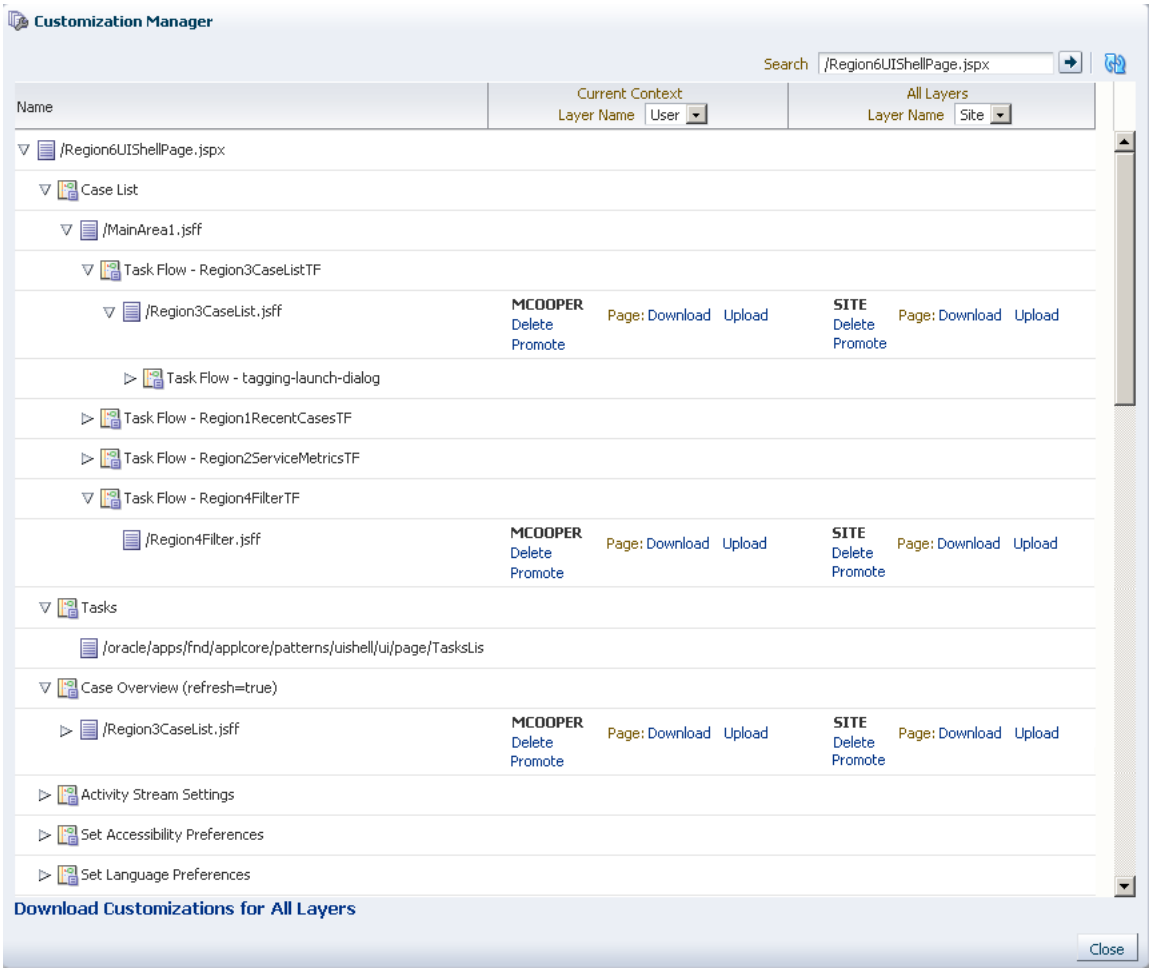
The Customization Manager is not available by default in your custom application. You can enable it by performing certain configurations. For more information, see Section 21.2.4, "How to Display the Customization Manager Add-On." If you configure your application to display the Customization Manager add-on, the Composer toolbar displays a Customization Manager button, as shown in Figure 18–42.

Figure 18–42 Customization Manager Button



Clicking the Customization Manager button invokes the Customization Manager dialog, shown in Figure 18–43.

Figure 18–43 Customization Manager Dialog



The Customization Manager displays Delete, Promote, Download, and Upload links against each customized page, fragment, or XML document in each layer. Users can specify a JSPX, JSFF, or XML file name in the search field to view customization details about objects in that file and manage those customizations. To search for a file, you must specify the absolute path to the file, for example, `/oracle/webcenter/portalapp/pages/home.jspx`.

Note: In case of a deeply nested task flow or a task flow with a cyclic dependency, Customization Manager displays a maximum depth of ten task flows.

Users can manage customizations in the current MDS context in which the application is running. The Current Context column displays options to manage customizations in the selected layer. The dropdown menu lists the layers in which the page or fragment was customized. This column is useful for all users who want to manage customizations they have made. The All Layers column displays options to manage customizations in all layers in the application. This column is useful for administrators who want to manage customizations made in all layers.

Notes:

- If you have configured a sandbox for your application, all Customization Manager operations are performed in the sandbox; for example, when a user uploads a customization document, it is stored in the sandbox until the user saves or discards changes made during that session.
- You can customize the Customization Manager to only display options of your choice. For example, you can show or hide the Download, Delete, or Upload links. Further, you can choose to show details for the current context only or for all layers only.

  You can make these settings using parameters on the Customization Manager task flow. By enabling parameter support on the task flow, you can enable users also to customize the Customization Manager at runtime. For more information, see Section 21.8, "Defining Property Filters."

This section describes the capabilities provided by the Customization Manager. It contains the following subsections:

- Download Customization Metadata
- Download Customization Metadata for All Customized Objects
- Upload Customization Metadata
- Delete Customization Metadata
- Promote Customization Metadata

Download Customization Metadata

A Download link enables users to download the customization document for the selected page, fragment, or XML document from the selected layer. This is useful if a user wants to edit customization metadata for a component or inspect the customization metadata. For example, if a page displays errors, the user can download the customizations, review them, and send them to Oracle Support Services, if required. This helps in diagnosing any customization issues on the page.

The Save dialog lets users select a location for saving the document. Users can edit this file locally and upload it to the site again. For information about uploading files, see Upload Customization Metadata.

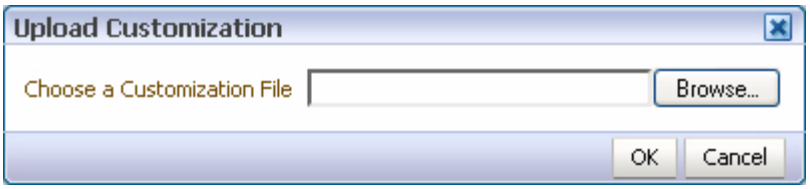
Download Customization Metadata for All Customized Objects

The Download Customizations for All Layers link at the bottom of the page enables users to download a ZIP file containing all customization documents shown in the Customization Manager.

Upload Customization Metadata

An Upload link enables users to upload a customization document for the selected artifact, such as page, fragment or XML document, in a specific layer. Figure 18–44 shows the Upload Customization dialog that lets users select the file they want uploaded.

Figure 18–44 Upload Customization Dialog



The artifact is then customized based on the metadata in the uploaded document.

Note: The `usesUpload` attribute on the `Form` tag on the page controls the display of Upload links in Customization Manager. If you set `usesUpload` to `false`, the Upload links will not work.

Delete Customization Metadata

A Delete link enables users to delete the customization document for the selected artifact from a specific layer.

Promote Customization Metadata

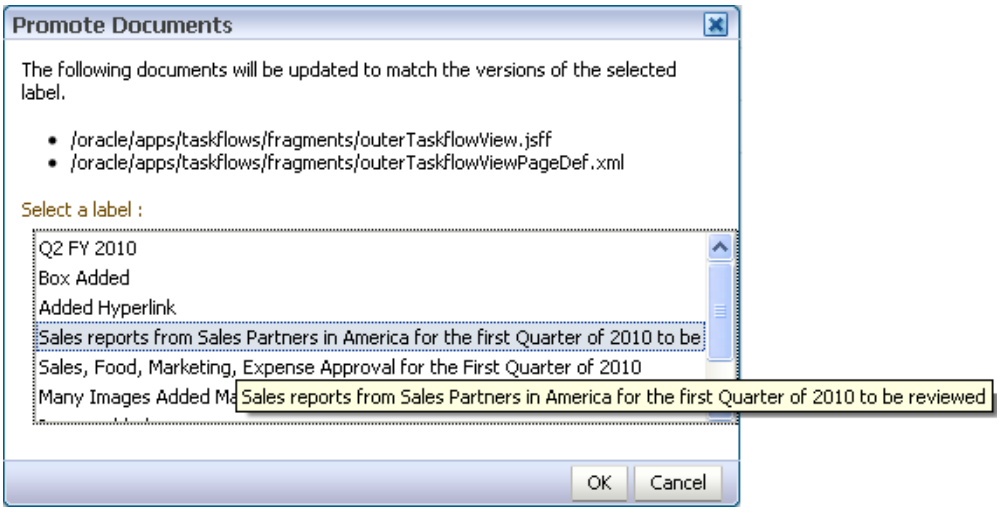
While customizing a page, users can save application customizations at any point and create a new label to save those customizations. A Promote link in the Customization Manager enables users to select a label and reset application customizations on the selected page or task flow to those from the label.

Notes: The Promote option is available only in sandbox-enabled applications.

For more information about the Save and Label option in Composer, see Section 18.4.13, "Create Labels On Saving Application Customizations."

The Promote link invokes the Promote Documents dialog, which lists all available labels and the files that will be promoted, as shown in Figure 18–45.

Figure 18–45 Select a Label Dialog

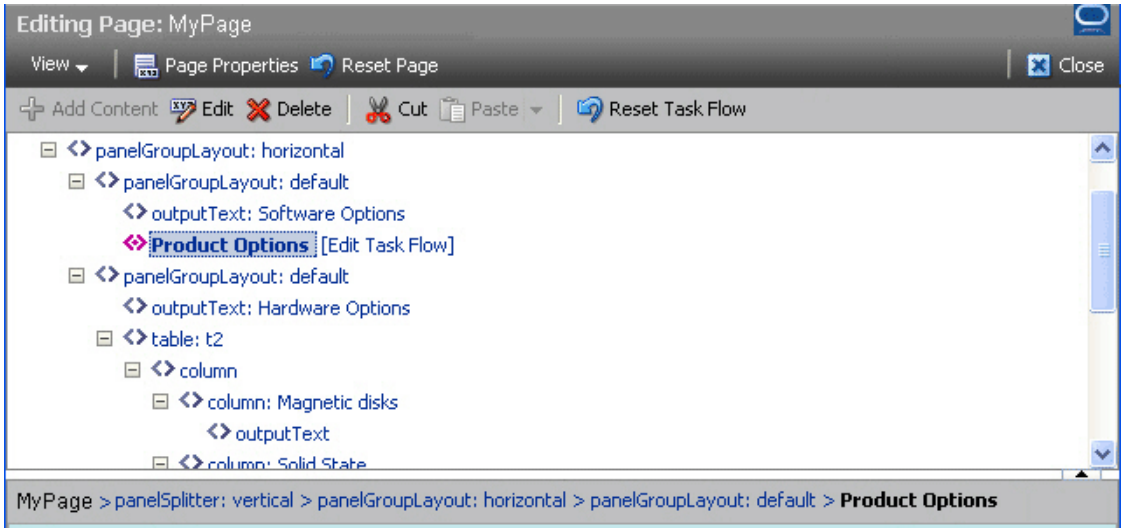


Users can select a label and promote that label to the sandbox for the selected artifact. This means that application customizations for the selected artifact are reset to those from the specified label. If a task flow is selected, then all page fragments and page definitions are promoted. If a page or fragment is selected, then the page or fragment and its page definition are promoted. Users can select a label from any available layer and promote customizations from that label. For example, if an application has layers MCOOPER, JDOE, and SITE, and a user working in the MCOOPER layer clicks the Promote link for a page, the Promote dialog lists all labels available in all three layers.

18.5 Editing Capabilities in Source View in Page Edit Mode

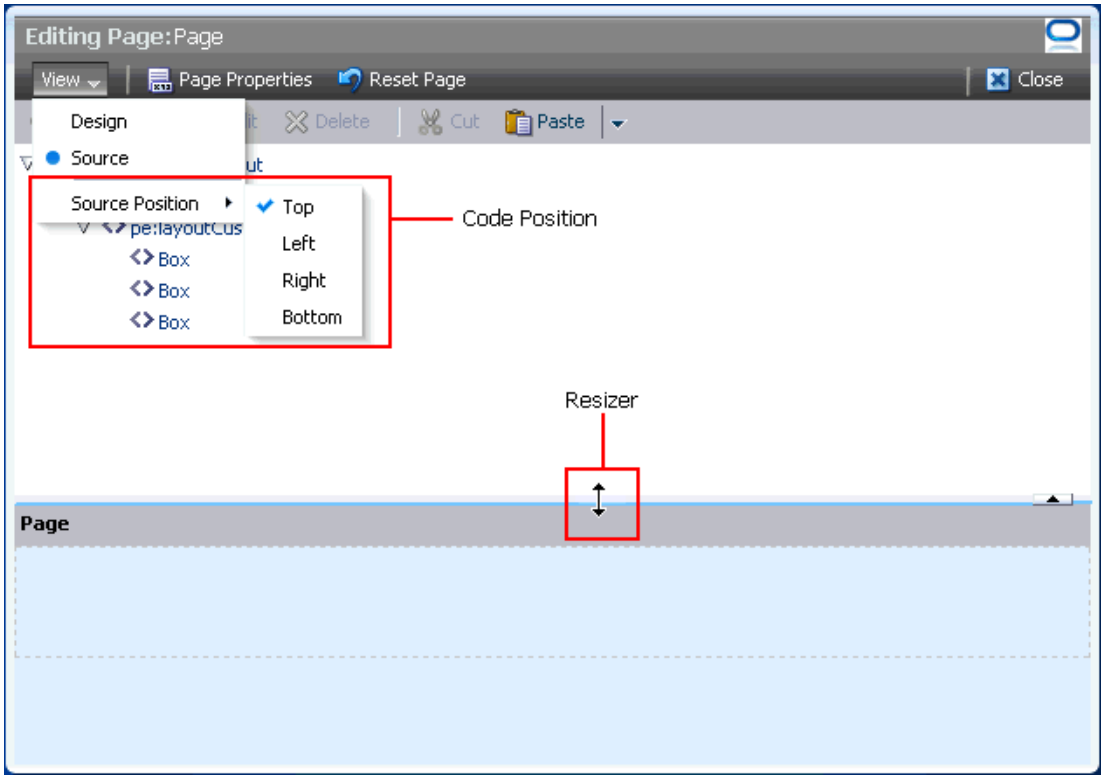
Source view (Figure 18–46) provides a WYSIWYG and a hierarchical rendering of page components in a component navigator. Add Content, Edit, Delete, Cut, and Paste controls are available on the Source view toolbar provide specific operations in Composer. In Source view, users can access and modify properties of components that are not otherwise selectable in Design view. For example, many ADF Faces components can be edited only in Source view. Users can also edit components within a task flow.

Figure 18–46 Source View of a Page



By default, the component navigator displays at the top of the page within Composer. Users can choose to display it at the bottom, left, or right using the **Source Position** option on the **View** menu (Figure 18–47). Users can also drag the border on the edge of the component navigator to alter its height or width.

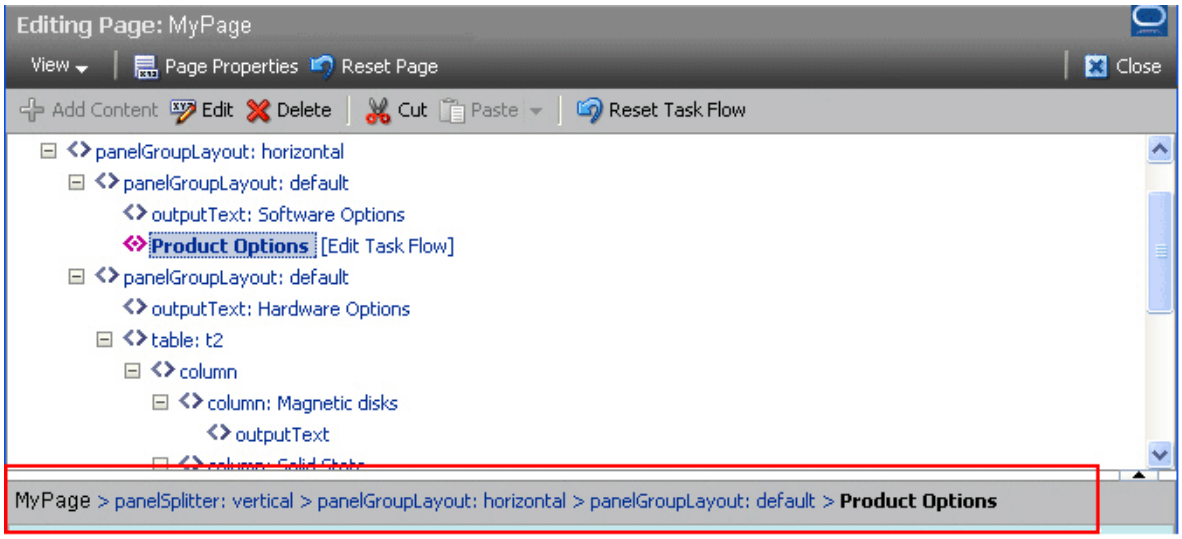
Figure 18–47 Source Position Option on the View Menu



At the root of the component navigator is the direct child of the `Page Customizable` component at design time. When a node is selected in the component navigator, the corresponding component is selected on the page. Similarly, when a component is selected directly on the page, the corresponding node is selected in the component navigator.

A gray area located between the component navigator and the page displays a bread crumb trail for the selected component (Figure 18–48). The bread crumb trail shows how the component is nested on the page. When no component is selected, the bread crumb bar displays just the page name. The container names are links that when clicked highlight the selected component.

Figure 18–48 Bread Crumbs for a Component Selected in Page Source View



Task flows that can be edited have Edit Task Flow links next to them. On clicking an Edit Task Flow link, the component navigator displays the hierarchy of components within the task flow's page or fragment. Users can edit this page or fragment and click Close to zoom out of the task flow. For more information, see Section 18.5.2, "Edit Content Inside a Task Flow." Similarly, declarative components display Open links that enable users to edit components inside the selected declarative component's facets.

Similar to Design view, in Source view also users can perform such tasks as adding components, editing page and component properties, changing page layout, and deleting components. Users can select a component on the page or in the component navigator and click the buttons on the Source view toolbar to perform these tasks. To edit or delete a component, users can also select options on the context menu for the component. In addition to these tasks, users can perform the following tasks in page Source view:

- Section 18.5.1, "Rearrange Page Content Using Cut and Paste Options"
- Section 18.5.2, "Edit Content Inside a Task Flow"
- Section 18.5.3, "Edit Content Inside a Declarative Component"
- Section 18.5.4, "Show or Hide Components"

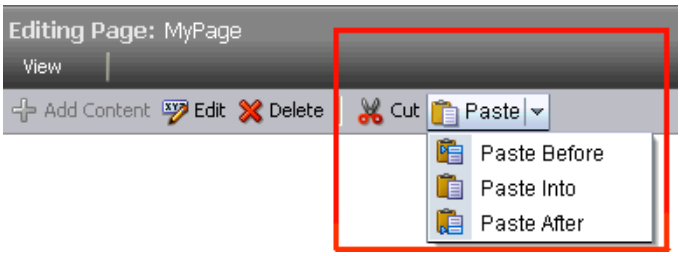
Notes:

- The **Save** button is provided on the Composer toolbar (both in Design view and Source view) only if the application is configured to use a sandbox. For more information about sandboxes, see Section 23.2, "Using Composer Sandbox."
- You can restrict users from performing all or some of these tasks by securing the application and applying customization restrictions on components. For more information, see Section 18.7, "Security and Composer."

18.5.1 Rearrange Page Content Using Cut and Paste Options

Users can rearrange components on a page using the Cut and Paste options on the Source view toolbar (Figure 18–49). A Paste menu enables users to paste a component into, before, or after any other component in the component navigator.

Figure 18–49 Cut and Paste Options on the Source View Toolbar



The Cut and Paste buttons are grayed out if the user selects a component that does not support a cut or paste operation. Cut and paste operations are not supported under the following conditions:

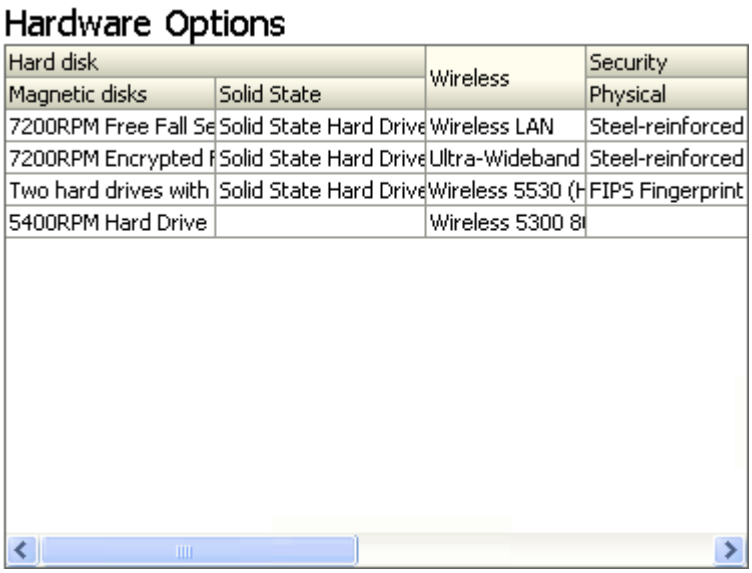
- The target component is not on the same page as the cut component.
- The target component is nested inside the cut component.
- The target component is located before or after a faceted component, that is, a component inside a facet.
- The parent of the target component is not customizable.
- The parent of the cut component is not customizable.

Notes: A component on which customization is restricted can be rearranged inside its parent container if the parent is customizable.

Example

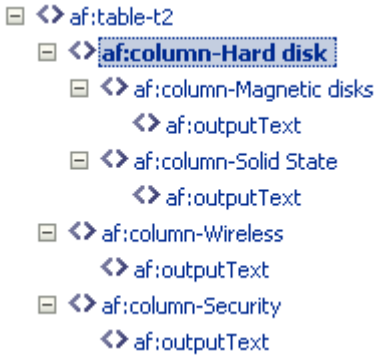
If your customizable page contains an ADF Faces `Table` component, then in Composer's Source view, users can resequence or move columns within or across tables and column groups. Consider a sample application created for selling computer accessories. The application home page has two tables containing details about the software and hardware offerings. The Hardware Options table contains four columns. The Hard disk column in turn contains the Magnetic disks and Solid State columns inside a column group, as shown in Figure 18–50.

Figure 18–50 Table in the Sample Application



Users can switch to Source view and resequence the Magnetic disks and Solid State columns within the Hard disk column or move them out of the column group and paste them next to any other column (the Wireless column, for example). Figure 18–51 shows the table structure in the component navigator.

Figure 18–51 Table Columns in Source View



Users can select a column and click **Cut**, then select another column and click **Paste Before** or **Paste After**.

18.5.2 Edit Content Inside a Task Flow

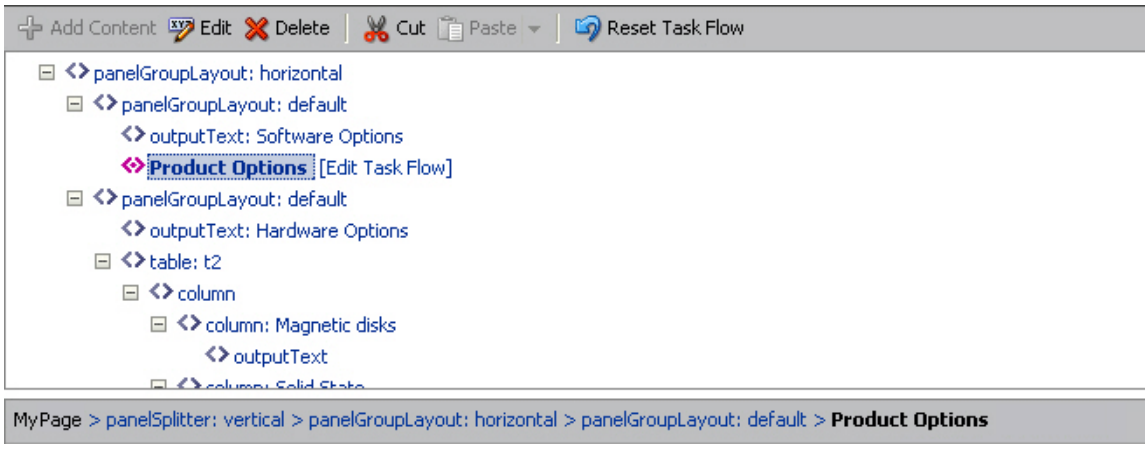
Oracle ADF Task flows consist of one or more views and each view in a task flow is associated with a page or page fragment. When you add a task flow to your page, the content on the fragment associated with the current view is displayed on the page. Composer enables users to edit components on the page or fragment used for the current view of a task flow. Since changes are made to the task flow's page or fragment, they are reflected in all places where the task flow's page or fragment is used.

The Source view provides an option to open a task flow and display only components on its page or fragment in the component navigator. Users can edit the page or fragment and close the task flow to navigate back to the page containing the task flow.

Users with Customize permission on a task flow can edit its page or fragment.

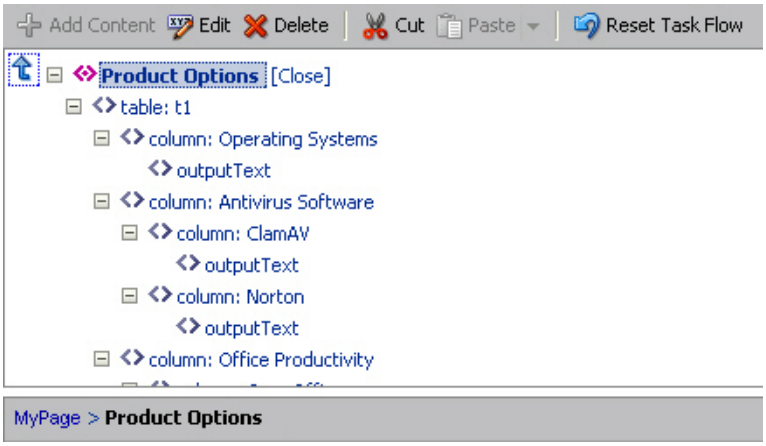
Each task flow in the component navigator displays an **Edit Task Flow** link next to it, as shown in Figure 18–52.

Figure 18–52 Edit Task Flow Link on a Task Flow



Clicking the **Edit Task Flow** link results in the Source view zooming in to display the task flow's page and its components, as shown in Figure 18–53.

Figure 18–53 Components Within a Task Flow

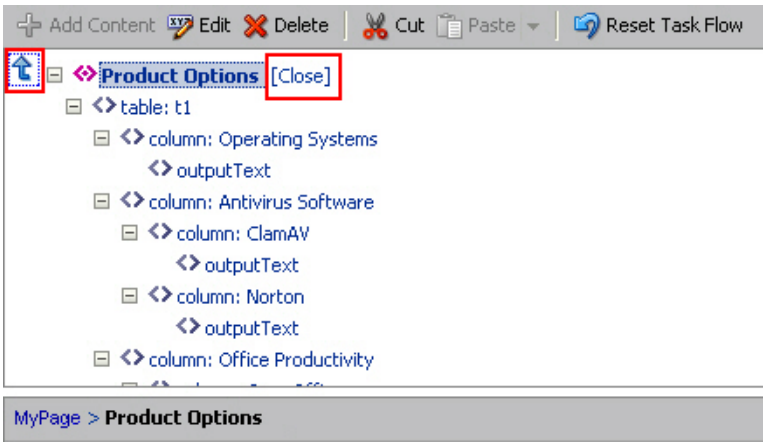


Users can select components on the task flow's page and edit them in Composer just like editing any other page components. However, Composer does not support moving components from one task flow to another.

If multiple users attempt to edit the same task flow at the same time, a concurrency warning appears in Composer that alerts each user to the others. However, this warning is displayed only if a sandbox is configured in the application. For information about how changes are saved in such cases, see "What Happens During Concurrent Edits".

Clicking the Up arrow or **Close** link next to an open task flow (Figure 18–54) results in the component navigator displaying the page containing the task flow.

Figure 18–54 Options to Zoom Out of a Task Flow

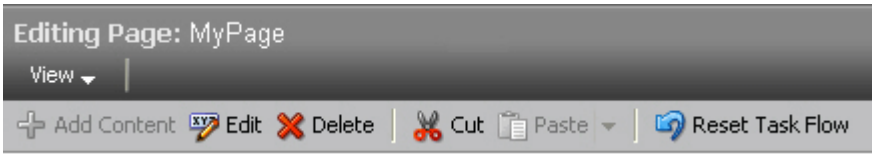


18.5.2.1 Reset Task Flow

The **Reset Task Flow** button on the Source view toolbar (Figure 18–55) invokes the Reset Task Flow dialog that provides options to remove application customizations made to a task flow and reset it to a previously-saved version or to its original out-of-the-box state. You can ensure that the task flow is reset to its state in a previously-saved label only if your application is configured to use a database store. The **Reset Task Flow** button is rendered only on zooming into a task flow.

> **Note:** If you disable the **Reset Page** button in your application, the **Reset Task Flow** button is also disabled. Users cannot reset task flows after they make changes. For more information, see Section 18.4.11, "Reset Page."

Figure 18–55 Reset Task Flow Button



Roll Back to a Specific Label

If your application is configured to use a database store, a new version of the task flow is generated each time a user saves application customizations. If a sandbox is configured for your application, by creating a label with the `SAVED_LABEL_%` convention you can ensure that Composer resets the task flow as follows:

- If there is only one label name following the `SAVED_LABEL_%` convention, then Composer resets the task flow to its state in that label.
- If there is no label name following the `SAVED_LABEL_%` convention, then Composer resets the task flow to its original, out-of-the-box state.
- If there are multiple labels following the `SAVED_LABEL_%` convention, then Composer resets the task flow to its state in the latest label among these.

To roll back to a specific label, you must have configured your application to use a sandbox. Specifically, the namespace of the task flow being customized must be defined in the `<metadata-namespaces>` section in the application's

`adf-config.xml` file. For more information, see Section 23.2.1, "How to Enable Composer Sandbox Creation."

Rolling back to a specific label is useful if your application has dependencies on the customization metadata and resetting the task flow to its original state may cause issues.

When a user resets a task flow, the following changes occur:

- Only the currently selected fragment is reset. Other fragments of the task flow are not reset.
- Nested task flows are not affected by the reset operation.
- When a task flow is reset, changes are reflected on all pages consuming this task flow.
- Components added to the task flow at runtime are removed. However, if the task flow's page or fragment uses a template, then components added inside `Panel Customizable` components on the template are not removed.

Example 18–2 shows the sample code to create a label with a `SAVE_LABEL_` prefix.

Example 18–2 Code to Create a Label Prefixed with SAVE_LABEL_

```
import oracle.mds.versioning.VersionHelper;
import oracle.mds.naming.Namespace;

. . .
long labelNumber = (long)(Math.random() * 1000000000);
String savedlabel = "SAVE_LABEL_" + labelNumber;

MDSInstance mdsInstance =
(MDSInstance)ADFContext.getCurrent().getMDSInstanceAsObject();

VersionHelper vh = VersionHelper.get(mdsInstance);

Namespace[] validNamespaces = new Namespace[1];
Namespace pageNamespace = Namespace.create(<pagePath or taskflow path>,
NamespaceRestriction.CUSTOMIZATIONS);

validNamespaces[0] = pageNamespace;
vh.createLabel(savedlabel, null, validNamespaces);
```

18.5.3 Edit Content Inside a Declarative Component

If the application page contains declarative components, users can edit these components in Composer. Editing declarative components is similar to editing task flows. For more information, see Section 18.5.2, "Edit Content Inside a Task Flow."

Figure 18–56 and Figure 18–57 show the Edit Component and Close options available on declarative components.

Figure 18–56 Edit Option On a Declarative Component

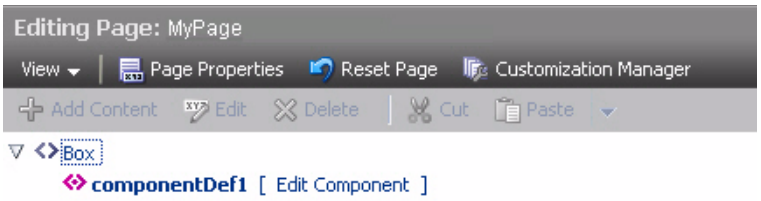
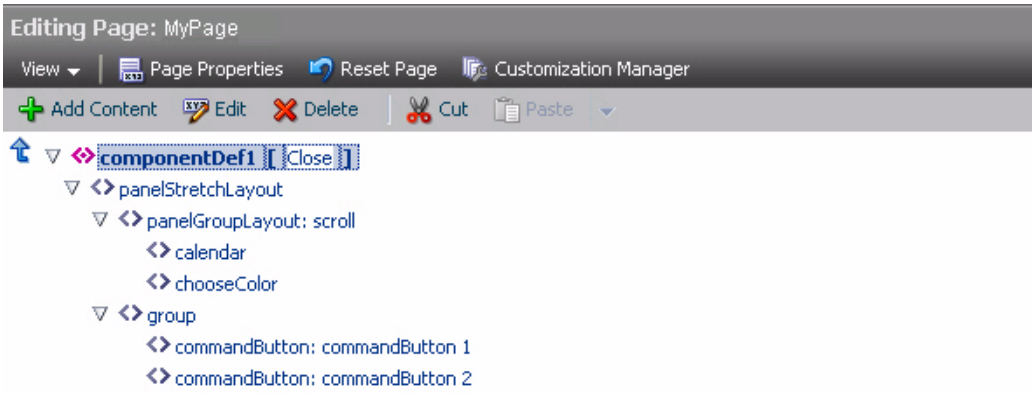


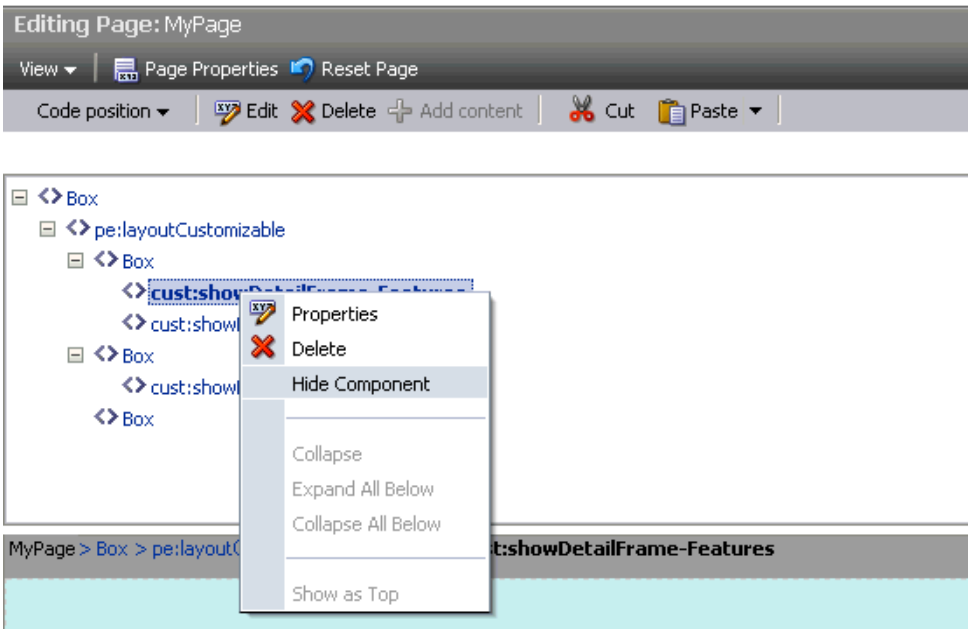
Figure 18–57 Close Option Displayed While Editing a Declarative Component



18.5.4 Show or Hide Components

To hide a component, right-click the component in the component navigator or on the page and choose Hide Component on the context menu (Figure 18–58).

Figure 18–58 Hide Component Option in Source View of the Page



A hidden component is not displayed on the page, but appears grayed out in the component navigator. The context menu for a hidden component displays a Show Component option. Choosing this option renders the component on the page again.

Note: Show or hide behavior is tied to the component's `rendered` property. When a user hides a component, its `rendered` property is set to `false` and vice versa. If the `rendered` attribute on a component has an EL value, then that value is lost on using the show or hide option. The value is set to `true` or `false`.

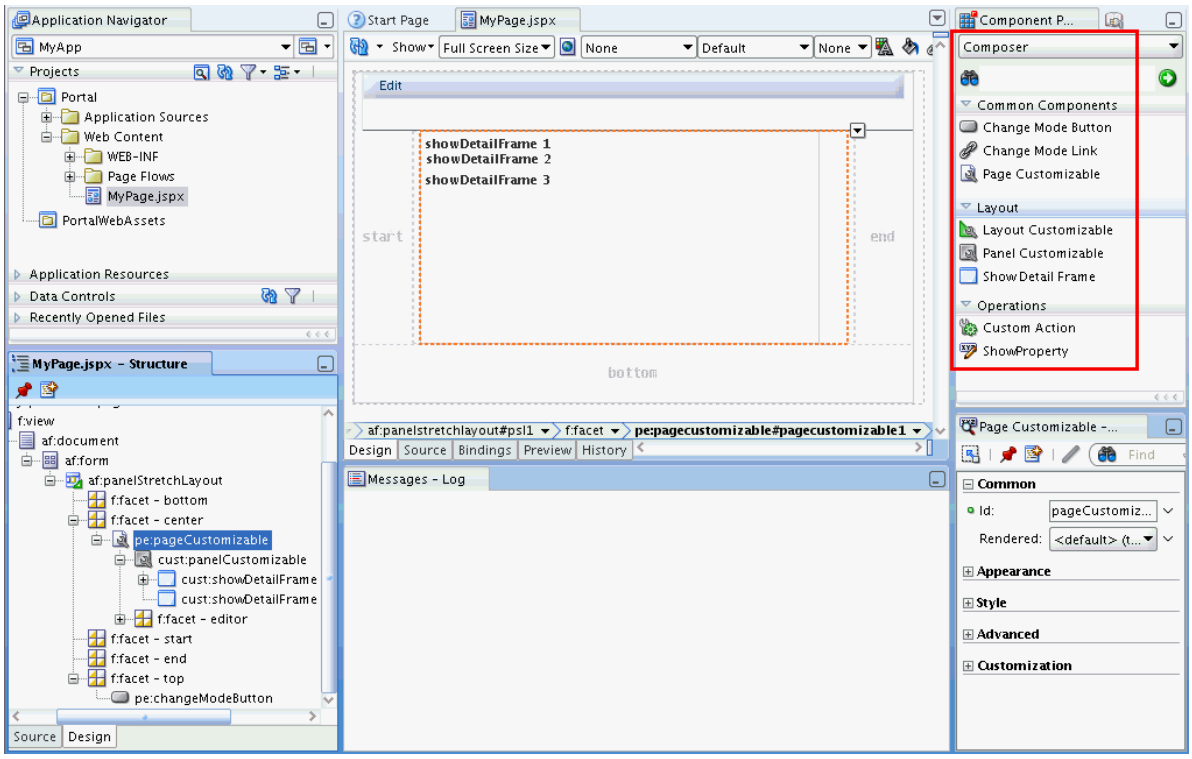
18.6 Composer Components

To make any JSPX document (`*.jspx`) editable at runtime, you must add Composer components to your page in Oracle JDeveloper at application design time.

Note: Composer works only with JSPX pages and ADF Faces. You cannot add these components to JSP pages. For information about adding Composer components to your page, see Section 20.1, "Designing Editable Pages Using Composer Components."

The Composer tag library (Figure 18–59) available from the Component Palette provides components that you can add to make a page editable.

Figure 18–59 Composer Tag Library in the Component Palette



Composer

****

This section provides an overview of the Composer components that are used to enable page editing. It contains the following subsections:

- Section 18.6.1, "Page Customizable"
- Section 18.6.2, "Change Mode Link and Change Mode Button"
- Section 18.6.3, "Panel Customizable"
- Section 18.6.4, "Show Detail Frame"
- Section 18.6.5, "Layout Customizable"

For more information about these components and other Composer components such as `Custom Action`, see Appendix B, "Composer Component Properties and Files."

18.6.1 Page Customizable

The `Page Customizable` component defines the editable area of a page. Within this area, you can edit component properties, add content to the page, arrange content, and so on.

Adding a `Page Customizable` component enables the runtime inclusion of Composer on the page. Users can edit pages in Composer using page-related controls available across the top of the page, **Add Content** buttons on components, and **Edit** icons on each editable page component, as shown in Figure 18–60.

Figure 18–60 Page Customizable Component

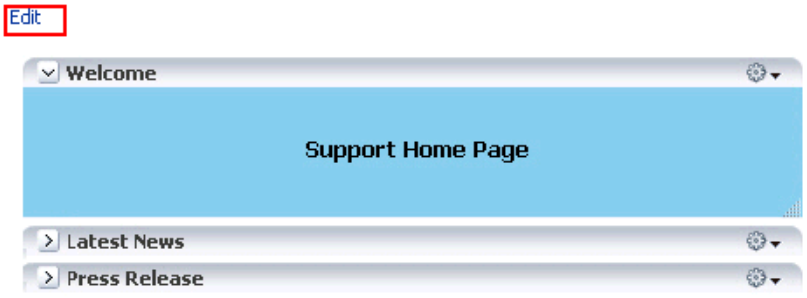


To enable runtime editing for multiple application pages in one operation, add a `Page Customizable` component to the ADF page template used for those pages. This avoids the need to manually add a `Page Customizable` to each page. For more information about adding the `Page Customizable` component, see Section 20.1.2, "How to Enable Runtime Customization Using a Page Customizable."

18.6.2 Change Mode Link and Change Mode Button

The `Change Mode Link` or `Change Mode Button` component provides an easy way to switch from View mode of the page to Edit mode. Figure 18–61 shows a `Change Mode Link` component on the page.

Figure 18–61 Change Mode Link Component



For more information about using `Change Mode Link` or `Change Mode Button`, see Section 20.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."

18.6.3 Panel Customizable

A `Panel Customizable` defines an area of the page onto which users can add components at runtime. Users can move or minimize `Show Detail Frame` components and portlets that are added as child components of a `Panel Customizable`. Users can also drag and drop these components into another `Panel Customizable` component on the page.

In Edit mode, the `Panel Customizable` component is rendered as a box with dotted lines. In fact, a `Panel Customizable` is referred to as a `Box` in the runtime Resource Catalog. An **Add Content** button appears on each `Panel Customizable` component on the page, as shown in Figure 18–62. You can use this button to open the Resource Catalog Viewer and add components within the `Panel Customizable`.

Figure 18–62 Panel Customizable Component

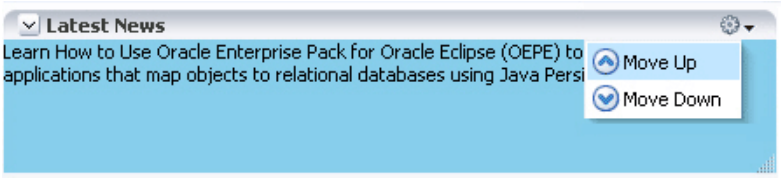


For more information, see Section 20.1.4, "How to Define Editable Areas of a Page Using Panel Customizable Components."

18.6.4 Show Detail Frame

A `Show Detail Frame` component renders a border or chrome around its child component along with a header that contains icons to enable users to perform some operations. The actions available on this menu enable users to move the component, along with its content, to new positions on the page (Figure 18–63). Users can also drag and drop `Show Detail Frame` components from one `Panel Customizable` component to another on the page. Note that a `Show Detail Frame` must be included inside a `Panel Customizable` component for it to be movable.

Figure 18–63 Show Detail Frame Component



A `Show Detail Frame` component enables the following actions:

- Collapse and expand the component
- Move content to different positions on the page
- If you add a task flow as a child component, then along with custom actions it enables task flow navigation using the Actions menu.
- If you add an ADF Faces `Rich Text Editor` as a child component, then it enables end users to edit and save text in the `Rich Text Editor`.

You can add your own UI controls to further customize the display of content by using facets of the `Show Detail Frame`. For information about using these facets, see Section 22.1, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets."

For information about adding this component to a page, see Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components."

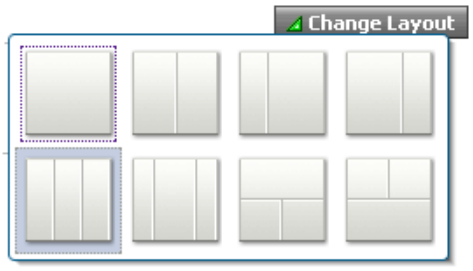
18.6.5 Layout Customizable

The `Layout Customizable` component is a container that enables end users to lay out its child components in several predefined ways (for example, two column, three column, and so on). You can design your page in such a way that all components on the page are enclosed in a `Layout Customizable` component. In such a case, the layout is applied to the entire page.

Access predefined layouts using the layout changer. By default, the layout changer displays as a small green icon both in page View mode and page Edit mode. By using

the `Layout Customizable` component attributes you can choose to display the layout changer as an icon, text, or icon and text. In addition, you can decide whether to show or hide the layout changer in View mode. Figure 18–64 shows a Layout Customizable component and the predefined layouts that display when users click the layout changer.

Figure 18–64 Layout Customizable Component



For more information, see Section 20.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable."

18.7 Security and Composer

In an application that is accessed by end users, application developers, and administrators, it may not be advisable to allow all users to perform all editing tasks. For example, you may want to allow end users to only customize their view of the page by performing tasks like rearranging components or hiding areas they do not want to see. On the other hand, application developers must be allowed to update content on the page, component properties, and so on. The ability to customize a page, component, or component attribute is inherited from security definitions at the tag, page, component, and attribute levels. However, you can override the default security definitions at various levels in keeping with your business requirement.

You can define application security on many levels, including on a page, an operation, a component, or a component attribute. There are several ways in which one can restrict application customizations on a component. Composer determines if a component can be customized by honouring these rules. For example, before users can customize components, the components' sources of security restrictions are queried to determine if application customization is allowed. If a component does not permit application customization, Composer considers it restricted; that is, Composer treats the component as if it were explicitly secured. Table 18–1 describes Composer behavior to reflect application customization restrictions.

Table 18–1 Customization Restrictions on Composer Components

Restriction	Behavior
The page (and its contents) is open for all customizations	Composer allows editing of components without restrictions.
The page is restricted	Composer displays but none of the options, for example **Page Properties** button and **Edit** icon, are accessible.
Some operations are restricted	Composer displays but the options corresponding to the restricted operations are disabled.
A component is restricted	The **Edit** icon is not rendered on the component and its properties cannot be edited.

Table 18–1 (Cont.) Customization Restrictions on Composer Components

Restriction	Behavior
A component's attributes are restricted	The restricted attributes are not displayed in Composer. Composer's API enables the developer to determine if a component's attributes are restricted or not.

Depending on the privileges granted while implementing security, users can perform different user and application customization tasks when they log in to the application.

> **Note:** In a secured application, it is recommended that you check all users' privileges and enable the Edit link or button on the page only for privileged users. Unauthenticated users who stumble into page Edit mode can change component properties.
>
> You can enable the Edit link or button for selected users by specifying an EL value for the `rendered` attribute on the `Change Mode Link` or `Change Mode Button` component. For more information, see Section B.1.2, "Change Mode Link and Change Mode Button."

This section describes the default security behavior of Composer components. It contains the following subsections:

- Section 18.7.1, "Page and Task Flow Security"
- Section 18.7.2, "MDS Customization Restrictions"
- Section 18.7.3, "Component Action-Level Security"

For information about overriding default security behavior, see Chapter 24, "Modifying Default Security Behavior of Composer Components."

18.7.1 Page and Task Flow Security

The application's `jazn-data.xml` file is the repository for page-level and task flow-level security information. Composer references this file and enables editing capabilities based on a user's privileges:

- A user with the Personalize privilege on a page or task flow can perform only the user customizations described in Section 18.3, "Customizing Capabilities in Page View Mode."
- A user with the Edit or Customize privilege can perform all runtime editing tasks such as adding content, editing component properties, and deleting components. Composer and WebCenter Portal Customizable Components do not differentiate between Edit and Customize privileges.

> **Notes:** Composer and WebCenter Portal Customizable Components support cascading of privileges with Grant being a super set of all privileges. A user with Grant privilege on a page or task flow is considered to have Edit, Personalize, and View privileges. A user with Personalize privilege is considered to additionally have the View privilege.

Table 18–2 explains Composer behavior based on page- and task flow-level privileges. Only those privileges that are relevant to Composer and WebCenter Portal

Customizable Components are listed in this table. The Grant privilege is not listed as it is a super set of all privileges. Users with the Grant privilege can perform all editing tasks.

Table 18–2 Mapping of Page or Task Flow Privileges to Composer Behavior

Privilege	Composer Behavior
Edit or Customize	Users can switch to Edit mode of the page, where Composer is invoked, and edit the page. Users with either the Edit or Customize privilege can perform all runtime editing tasks. With the Edit or Customize privilege on a page or task flow, users can: ■ Add content ■ Edit component properties ■ Rearrange content ■ Delete components in Composer ■ Personalize or customize a portlet ■ Move a portlet or task flow in View mode. This is persisted as a user customization. ■ Expand and collapse a task flow or portlet in View mode. This is persisted as a user customization. ■ Resize a column of a table in a task flow. This is persisted as a user customization. ■ Reset the page to its original state ■ Delete components in View mode Note: You can perform all or some of these tasks depending on whether you have page-level or task flow-level privileges. If users do not have the Edit or Customize privilege on a page but try to edit it, a message appears stating that they do not have permission to do so. If users do not have the Edit or Customize privilege on a task flow, the Edit option is not displayed for the task flow in Source view.
Personalize	In View mode, users with a Personalize privilege on the page can: ■ Rearrange content ■ Personalize a portlet ■ Move a portlet or task flow in View mode. This is persisted as a user customization. ■ Expand and collapse a task flow or portlet in View mode. This is persisted as a user customization. ■ Resize a column of a table in a task flow. This is persisted as a user customization. ■ Delete components in View mode **Note**: Having Personalize permission does not enable users to perform portlet customizations. If users without Personalize, Edit, or Customize permission try to edit a page, a message appears stating that they do not have permission to do so.
View	Users with the View privilege can only view the page or task flow, but not perform user or application customizations.

18.7.1.1 Task Flow Security

When you add a task flow to a customizable page, Composer provides options for editing the task flow and components on the task flow's page. In a secured application, Composer provides editing capabilities based on the privileges provisioned on the customizable page and the task flow. For more information about task flow privileges and Composer's behavior, see Table 18–2.

On an application page, components can be located directly on the JSPX page or on page fragments inside shared components such as task flows. Restrictions on components inside a task flow are derived from the page fragment. To understand task flow security better, consider an example of a page containing a task flow, which in turn contains another task flow, as shown in Figure 18–65.

Figure 18–65 Structure of a Page Containing Nested Task Flows



Components on this page inherit security as follows:

- `Component1`, `Component2`, and `Region3` inherit security definitions from `Document.jspx`.
- `ComponentA`, `ComponentB`, and `RegionC` inherit security definitions from `region3.jsff`.
- `ComponentX` and `ComponentY` inherit security definitions from `regionC.jsff`.

Task flows do not support cascading of permissions. That is, page permissions are not inherited by components inside a task flow. However, users who do not have Edit or Customize permission on the page cannot customize task flows on the page.

For example, if you grant Edit or Customize permission on `Document.jspx` and `regionC.jsff` but not on `region3.jsff`, then users can customize `Component1`, `Component2`, `RegionC`, `ComponentX`, and `ComponentY`, but not `ComponentA`, `ComponentB`, and `RegionC`.

In a secured application, task flow permissions are stored in the application's `jazn-data.xml` file. The list of available actions for a task flow is defined by the task flow permission class, `oracle.adf.controller.security.TaskFlowPermission`. The permission

class defines task flow-specific actions that it maps to the task flow's operations. By default, only View permission is provisioned on task flows. To enable users to edit a task flow at runtime, you must ensure that Customize permission is granted on the task flow. For more information, see Section 20.3.1.1, "Considerations for Adding Task Flows" **and Section 24.6, "Implementing Task Flow Security."**

For more information about task flows and their security behavior, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

18.7.2 MDS Customization Restrictions

Framework applications have a default MDS configuration that restricts application customization on all application objects. To enable runtime page editing, you must make this default restriction inactive by adding a `Page Customizable` component to the page. A `Page Customizable` component enables application customization on all components under it. However, it does not enable customization on components of a nested page or fragment. For example, if the `Page Customizable` is used in a page template and has a `Facet Ref` inside it, then customization is not enabled by default on the components inside the `Facet Ref`.

You can enable application customization on a set of attributes for the component using MDS type-level restrictions or instance-level restrictions. Type-level restrictions are applicable to a specified component type across instances. At runtime, attributes for which you have enabled customization are shown as editable properties in Composer, and restricted attributes are not displayed in the Component Properties dialog for the selected component. For information, see Section 24.1, "Applying Component-Level Restrictions by Defining Customization Policies."

18.7.3 Component Action-Level Security

`Panel Customizable` and `Show Detail Frame` components enable the placement of restrictions on individual supported actions. For example, one can specify a restriction on whether the current user is allowed to minimize the `Show Detail Frame`.

It is left to you to enforce restrictions on the actions on a component. You can specify restriction on component actions in `adf-config.xml`. If a restriction is specified and applicable to the current user, the `Panel Customizable` or `Show Detail Frame` does not render the action.

For information about applying action-level restrictions, see Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."

19

Enabling Runtime Creation and Management of Pages

This chapter describes how to use the Page service to create and manage pages at runtime. It includes the following sections:

- Section 19.1, "Introduction to Page Creation and Management"
- Section 19.2, "Creating Pages and Task Flows"
- Section 19.3, "Defining Values for the Page - Create New Task Flow Parameters"
- Section 19.4, "How to Create Task Flow View Pages"
- Section 19.5, "Managing Pages"
- Section 19.6, "Introduction to Custom Styles and Templates"
- Section 19.7, "Customizing Page Service Views"
- Section 19.8, "Advanced Information for the Page Service"
- Section 19.9, "Page Service Samples"

19.1 Introduction to Page Creation and Management

The Page service enables you to create new pages and task flows in your application at runtime. You can base your pages on default or custom styles and templates. You can create them with either the Page - Create New task flow or the Page service APIs.

After you create pages or task flows, users can view and manage them with either the Page service data control or the Page service APIs. The APIs provide a means of defining a work area within the application, called a scope. Scopes are useful for categorizing custom pages that are of interest to a specific team or community (similar to a Group Space in Oracle WebCenter Portal: Spaces).

Table 19–1 describes the developer tools included with the Page service.

Table 19–1 Page Service Developer Tools

Tool	Description
Page - Create New task flow	A task flow for creating pages or task flows at runtime.

Table 19–1 (Cont.) Page Service Developer Tools

Tool	Description
PageServiceDC data control	A data control for viewing or deleting information about listed pages and task flows at runtime. **PageServiceDC** is available in design time after you add the Page service libraries to your application. By default, the Page service libraries are added to the application when you use the WebCenter Portal application template or when you add the Page - Create New task flow.
Page service APIs	APIs for creating and managing pages and task flows at runtime.

The Page service is integrated with many Oracle WebCenter Portal: Services, such as the Links, Search, and Tags services. You can track the most recent changes in pages with the Recent Activities service. Because it is preconfigured to work with the Recent Activities service, the Page service automatically produces the information that the Recent Activities service uses to display the most recent additions, changes, or deletions in pages.

For more information about the services at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

19.2 Creating Pages and Task Flows

You can create pages and task flows at runtime with either the Page - Create New task flow or the Page service APIs.

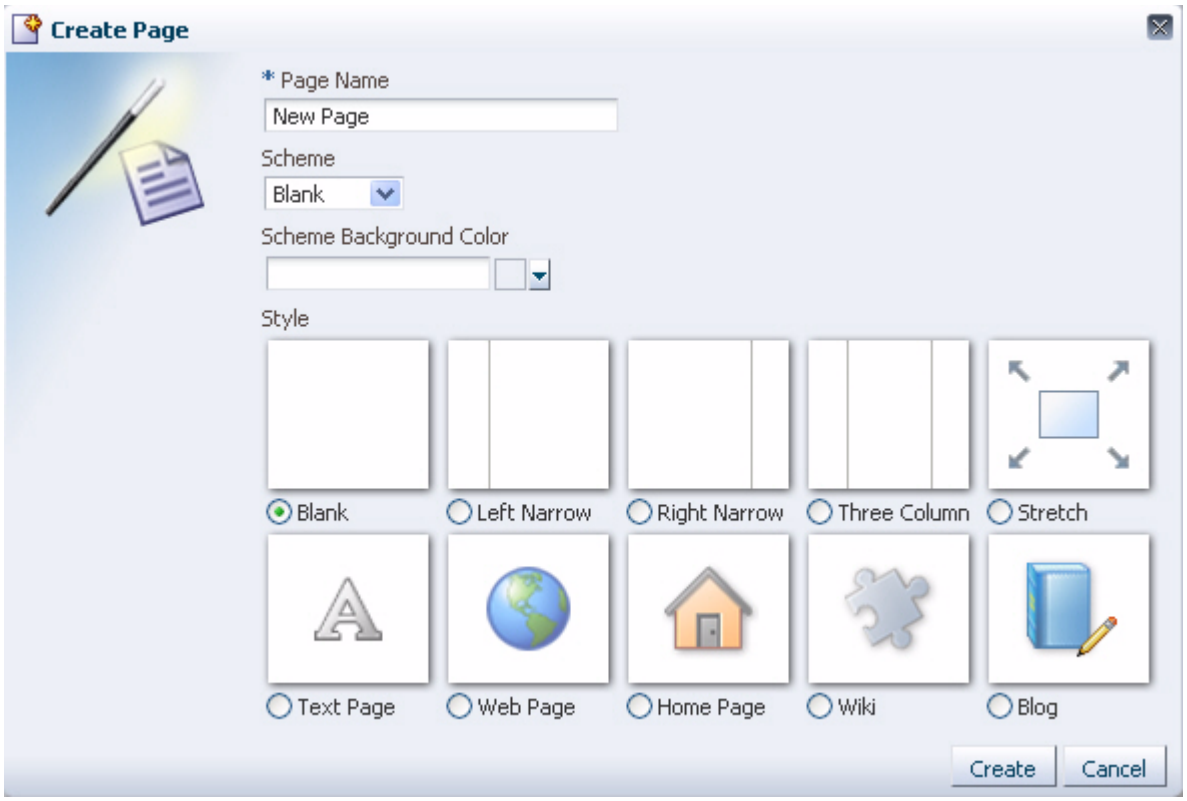
> **Note:** The Page - Create New task flow provides a means of creating pages and task flows and editing task flow parameters. To execute operations, such as copy, rename, create scope, and so on, you must use the Page service APIs along with the Page - Create New task flow. The Page service APIs provide the flexibility of customization within the Page service.

The Page - Create New task flow enables you to invoke the Create Page dialog at runtime. The Create Page dialog in turn enables users to create pages based on predefined styles, schemes, and templates.

Page styles, schemes, and templates provide both a default page structure that describes the areas where you can place content (that is, the page layout) and a background color and image that contribute to a page look and feel. You can select the page style, scheme, scheme background color, and template when you create a page. You can additionally enhance the usefulness and presentability of a page using page layout components. These include an in-place HTML text editor, images, layout boxes, hyperlinks, and so on.

Figure 19–1 shows the Create Page dialog at runtime.

Figure 19–1 Create Page Dialog with Default Styles



Some page styles include properties that suggest a particular use for the page. For example, the Web Page style includes a configurable property for specifying a URL. Among its many uses, the Web Page style provides a means of embedding wiki or blog pages and exposing external Web content within your application.

In many cases, you can switch the page scheme, scheme background color, and layout when you revise a page. You can also start with a blank page and create layout, look, and feel from the start.

At runtime, you can view a list of pages created through the Page - Create New task flow and create, copy, update or delete those pages.

For more information, see the section "Working with Page Layouts, Styles, and Schemes" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

19.2.1 How to Create Pages

This section describes how to add the Page - Create New task flow to your application to enable the creation of pages at runtime. It contains the following subsections:

- Section 19.2.1.1, "How to Add the Page - Create New Task Flow"
- Section 19.2.1.2, "Setting Security for the Page Service"
- Section 19.2.1.3, "How to Create Pages at Runtime"
- Section 19.2.1.4, "Structure of Pages Created at Runtime"
- Section 19.2.1.5, "How to Access Pages Created at Runtime"

19.2.1.1 How to Add the Page - Create New Task Flow

To add the Page - Create New task flow to your WebCenter application:

1. Follow the steps described in Section 5.3, "Adding Pages to a Portal."
2. Open the customizable page on which you want to add the service.
3. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
4. Drag and drop the **Page - Create New** task flow onto your page.
5. When prompted, select **Region** as the way to create the task flow.
6. Click **OK**.
7. Save your page.
8. Run your application to see that Create Page is now on the page.
9. Click **Create Page**.

   The Create Page dialog opens (see Figure 19–1).
10. Enter a page name, then select the scheme, the scheme background color, and the template.
11. Click **Create**.

    A page is created based on the selected information. For more details on where the page is created, see Section 19.2.1.4, "Structure of Pages Created at Runtime."

Note: After you add the task flow, you can edit task flow parameters. For information about Page - Create New task flow parameters, see Section 19.3, "Defining Values for the Page - Create New Task Flow Parameters."

For detailed information about pages, such as how to delete pages created at runtime, see Section 19.5, "Managing Pages."

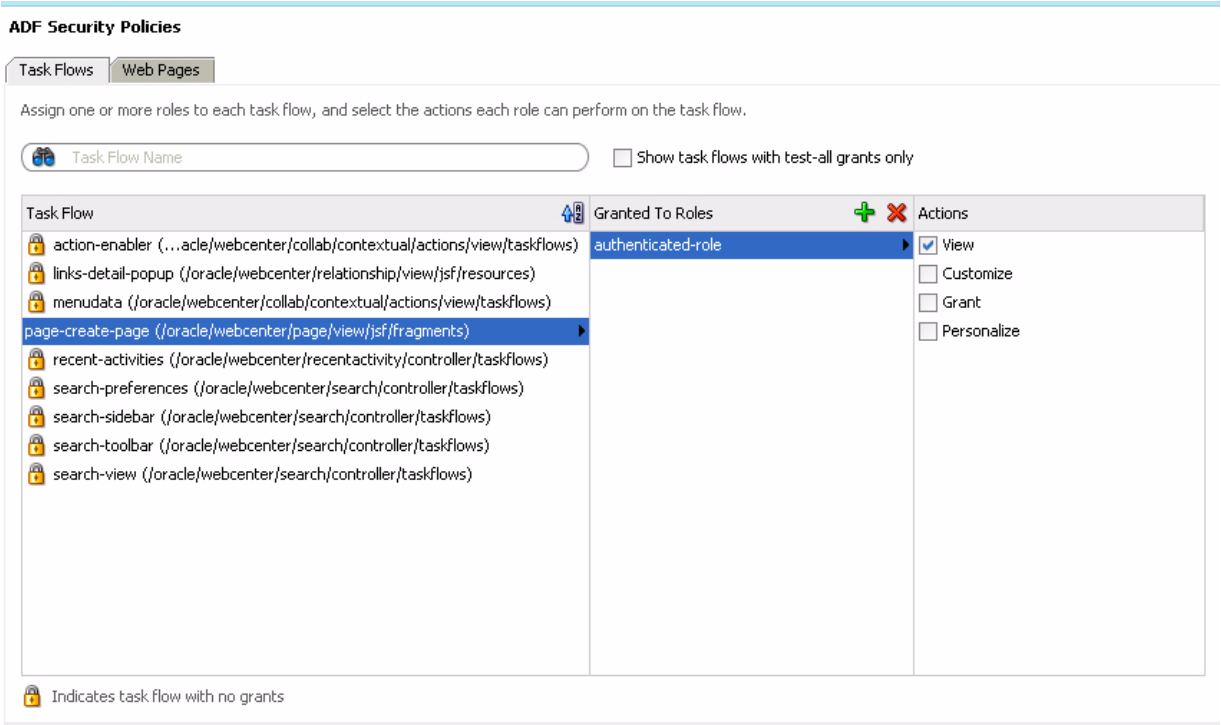
19.2.1.2 Setting Security for the Page Service

The Page service does not require its own, specific configuration of ADF security. It follows the security set up for the Framework application. In a non-secured application, all pages created in the Create Page dialog appear in the data control, viewable by everyone. In a secured application, users see only the pages they create or the pages to which they are granted privileges.

To secure your application and the Page - Create New task flow:

1. Follow the steps in Section 7.2.1.1, "Implementing Security for Services."
2. After you add the **Page - Create New** task flow to an application page, double-click the `jazn-data.xml` file in Application Resources to open the ADF Security Policies Editor.
3. If necessary, click the **Task Flow** tab to bring it forward.
4. In the **Task Flow** column, select **page-create-new** task flow (Figure 19–2).

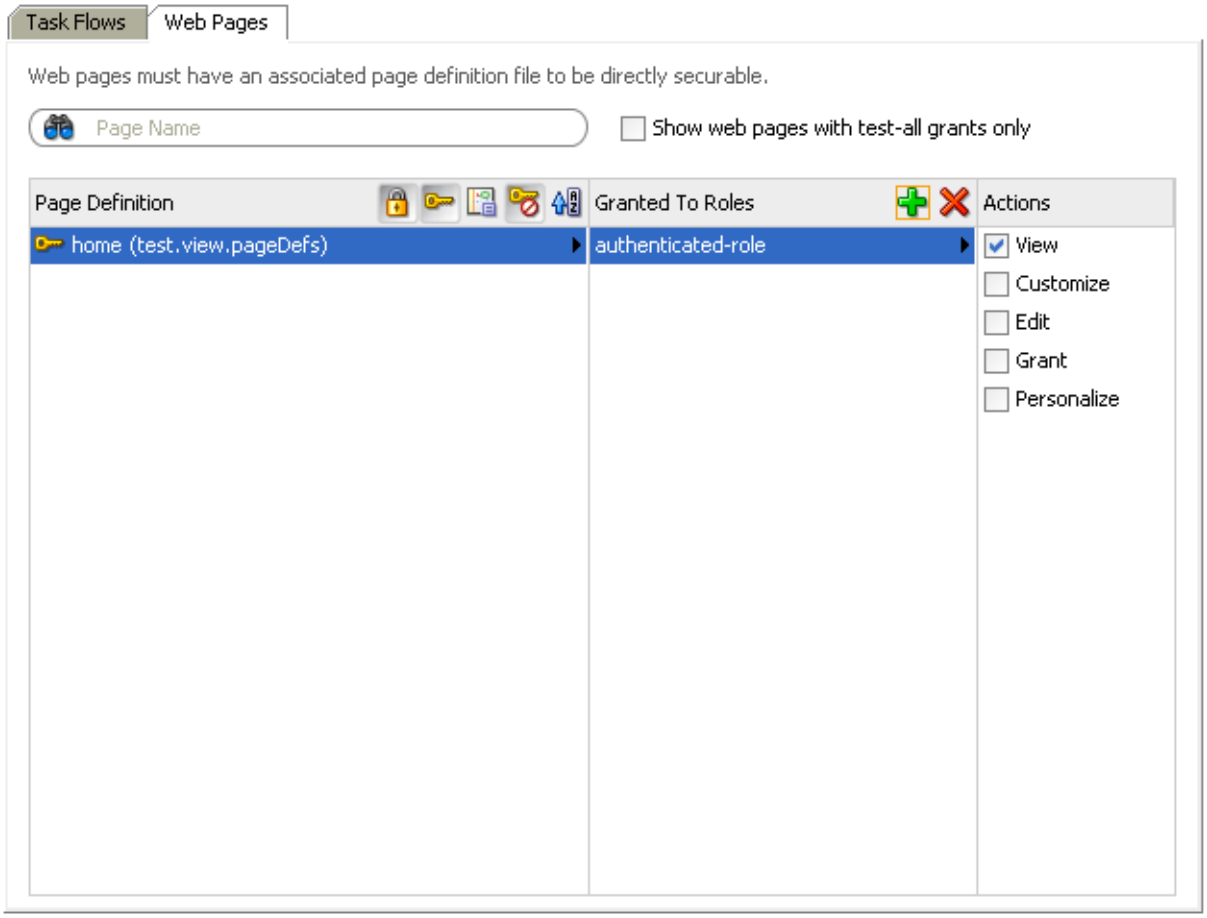
Figure 19–2 Adding Privileges to the Page - Create New Task Flow



5. Under **Granted to Roles**, click the **Add** icon to add a user role on which to grant task flow access privileges.
6. After you have added user roles, select a role in the ADF Security Policies dialog and then, under Actions, select the privileges to apply to the selected role.
7. Click the **Web Pages** tab to apply security to the page on which you added the Page - Create New task flow (Figure 19–3).

   Follow the same procedure outlined in steps 3 through 6: select a page, add roles, apply privileges.

Figure 19–3 Adding Privileges to Page with the Page - Create New Task Flow



Note: The **PageServiceDC** data control does not require any security setting for a secure or non-secure application.

19.2.1.3 How to Create Pages at Runtime

For detailed information, see the chapter "Introducing the Page Service and Oracle Composer" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

19.2.1.4 Structure of Pages Created at Runtime

The pages you create at runtime using the Page - Create New task flow or the Page service APIs are stored in the `mds` folder, located under your application root. Table 19–2 provides examples of the default formats used for page and task flow file names.

Table 19–2 Default File Name Formats

File Type	Format
Page	`Page<N>.jspx` (value of N starts from 1) For example: `Page1.jspx`, `Page2.jspx`, `Page<N>.jspx`
Page Definition	`Page1PageDef.xml`, `Page2PageDef.xml`, `Page<N>PageDef.xml`

Table 19–2 (Cont.) Default File Name Formats

File Type	Format
Task Flow View pages	`Page1.jsff`, `Page2.jsff`, `PageN.jsff`
Task Flow View Page Definitions	`Page1PageDef.xml`, `Page2PageDef.xml`, `Page<N>PageDef.xml`
Task Flow Definitions	`Page1.xml`, `Page2.xml`, `Page<N>.xml`

When you use the Page service APIs to create pages, you can provide a custom value for the file name format. For example: myPage<N>.jspx (value of N starts from 1)

The following examples show the structure of a page (`Page1.jspx`) and its associated page definition (`Page1PageDef.xml`) created at runtime by the user `user1`.

Example 19–1 Page Created Under a Default Scope

```
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_
296c916a23ed\user\user1\Page1.jspx
```

Example 19–2 Page Definition Created Under a Default Scope

```
mds\pageDefs\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_
296c916a23ed\user\user1\Page1PageDef.xml
```

Example 19–3 Page Created Under a Custom Scope

```
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx
```

Example 19–4 Page Definition Created Under a Custom Scope

```
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml
```

In Example 19–4, the `scopeGUID` is a unique ID assigned to a specific scope by the Page service.

Example 19–5 Task Flow View Page Created Under a Default Scope

```
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_
296c916a23ed\user\user1\Page1.jspx
```

Example 19–6 Task Flow Definition Created Under a Default Scope

```
mds\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_
296c916a23ed\user\user1\Page1.xml
```

Example 19–7 Task Flow View Page Definition Created Under a Default Scope

```
mds\pageDefs\oracle\webcenter\page\scopedMD\s8bba98ff_4cbb_40b8_beee_
296c916a23ed\user\user1\Page1PageDef.xml
```

Example 19–8 Task Flow View Page Created Under a Custom Scope

```
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.jspx
```

Example 19–9 Task Flow Definition Created Under a Custom Scope

```
mds\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1.xml
```

Example 19–10 Task Flow Page Definition Created Under a Custom Scope

```
mds\pageDefs\oracle\webcenter\page\scopedMD\<scopeGUID>\Page1PageDef.xml
```

In Example 19–10, the `scopeGUID` is a unique ID assigned to a specific scope by the Page service.

19.2.1.5 How to Access Pages Created at Runtime

To access pages with a default scope, enter the following URL:

```
http://<server>:<port>/<app name>/faces/oracle/webcenter/page/scopedMD/s8bba98ff_
4cbb_40b8_beee_296c916a23ed/user/<user guid>/Page1.jspx
```

To access pages with a custom scope, enter the following URL:

```
http://<server>:<port>/<app name>/faces/oracle/webcenter/page/scopedMD/<scope
guid>/Page1.jspx
```

19.3 Defining Values for the Page - Create New Task Flow Parameters

This section describes how to access and define values for Page - Create New task flow parameters. It contains the following subsections:

- Section 19.3.1, "How to Access Page - Create New Task Flow Parameters"
- Section 19.3.2, "Setting Scope in a Page - Create New Task Flow"
- Section 19.3.3, "Setting an Outcome Parameter"
- Section 19.3.4, "Specifying Styles"
- Section 19.3.5, "Specify an ADF Template"
- Section 19.3.6, "Showing a Command Link"
- Section 19.3.7, "Displaying an Image"
- Section 19.3.8, "Customizing the Label"

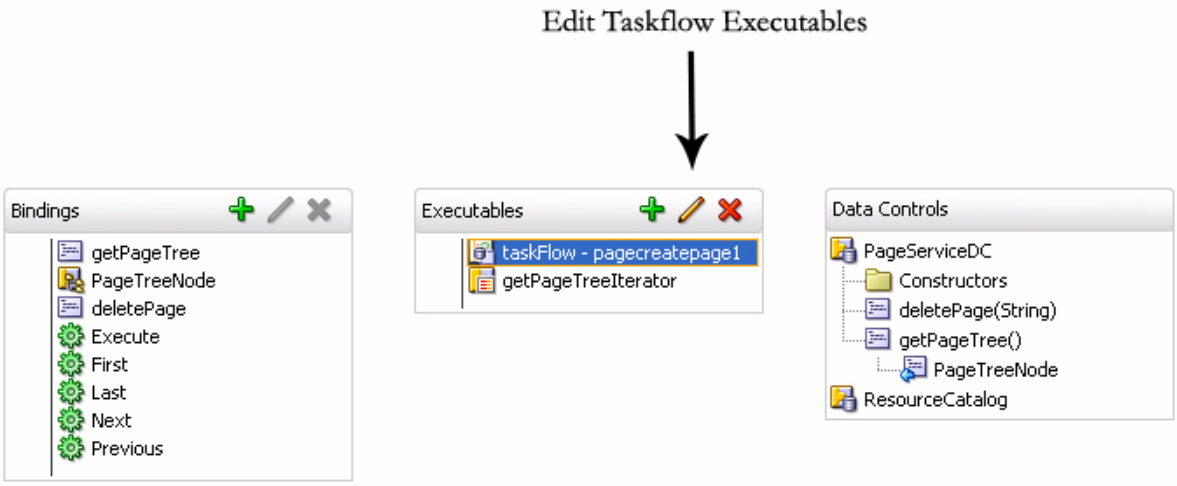
19.3.1 How to Access Page - Create New Task Flow Parameters

To access Page - Create New task flow parameters:

1. Go to the Application Navigator and right-click the `.jspx` page with the Page - Create New task flow.
2. Select **Go to Page Definition**, and scroll to select the **Create Page** task flow.
3. Click the **Edit** (pencil) icon to edit any of the parameters.

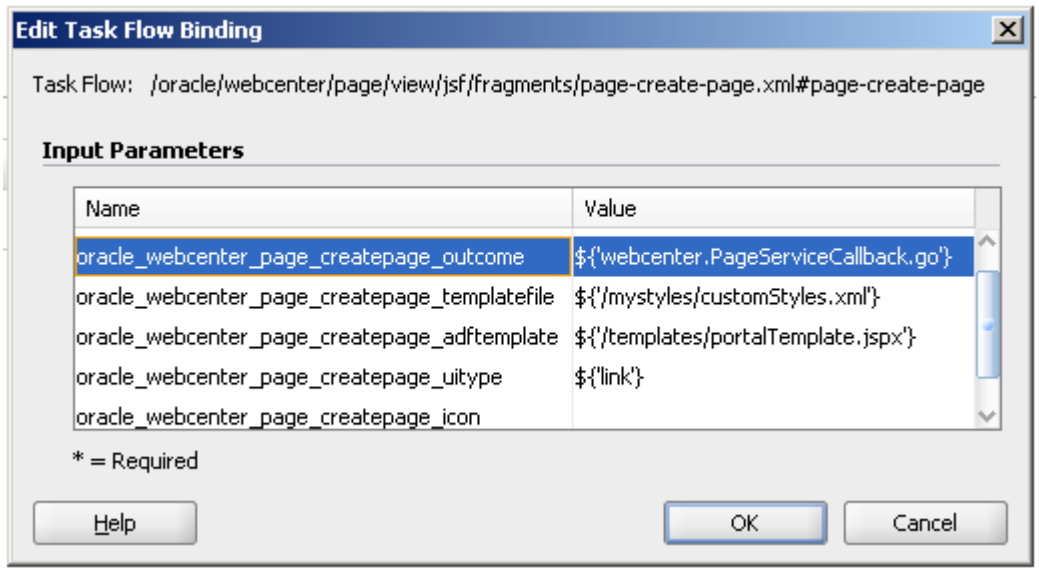
   The Page Data Binding Definition in the design view is shown in Figure 19–4.

Figure 19–4 Page Data Binding Definition



The **Edit Task Flow Binding** dialog opens, as shown in Figure 19–5.

Figure 19–5 Edit Task Flow Binding for the Page - Create New Task Flow



The following optional Page - Create New parameters are available:

- `scopename`

  For information, see Section 19.3.2, "Setting Scope in a Page - Create New Task Flow."

- `outcome`

  For information, see Section 19.3.3, "Setting an Outcome Parameter."

- `templatefile`

  For information, see Section 19.3.4, "Specifying Styles."

- `adftemplate`

For information, see Section 19.3.5, "Specify an ADF Template."

- `uitype`

  For information, see Section 19.3.6, "Showing a Command Link."

- `icon`

  For information, see Section 19.3.7, "Displaying an Image."

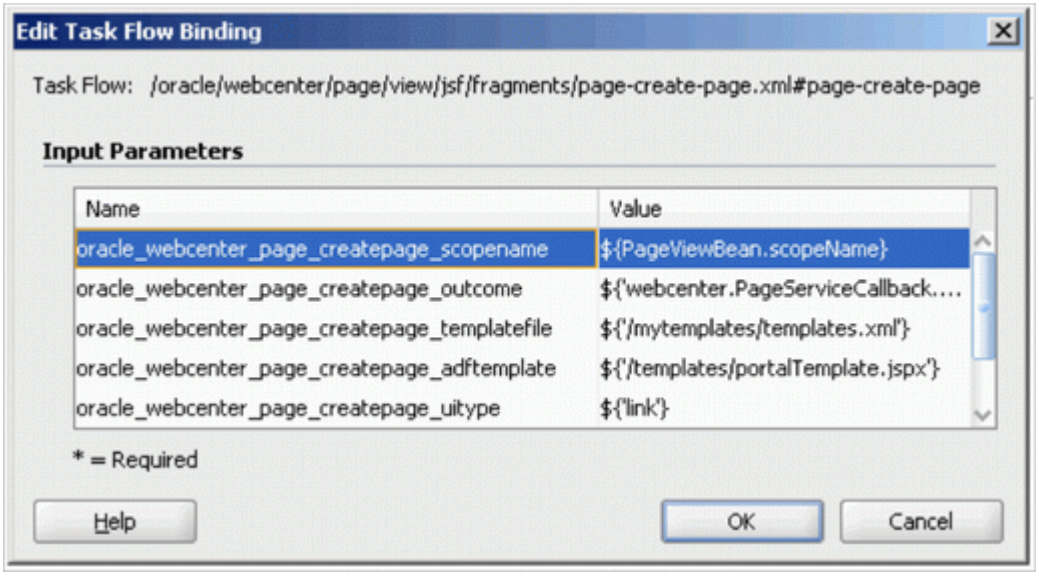
- `label`

  For information, see Section 19.3.8, "Customizing the Label."

19.3.2 Setting Scope in a Page - Create New Task Flow

By default, the pages and task flows you create using the Page - Create New task flow are stored in the default scope. To assign a specific scope to the Page - Create New task flow, the scope name must be assigned to the `oracle_webcenter_page_createpage_scopename` parameter, as shown in Figure 19–6.

Figure 19–6 oracle_webcenter_page_createpage_scopename Parameter



In Figure 19–6, `PageViewBean` is a custom managed bean that is invoking the `getScopeName()` method to fetch the value of `scopeName`. It uses the Page service API. By doing this, the Page - Create New task flow creates all pages and task flows under the specified scope, rather than default scope.

The following example code highlights only the creation of scope. If a scope already exists, then users can get the scope name from the `Scope` object.

Example 19–11 Creating a Scope for Application Pages

```
import oracle.webcenter.framework.service.Scope;
import oracle.webcenter.framework.service.ScopeAlreadyExistsException;
import oracle.webcenter.framework.service.ServiceContext;

public class PageViewBean {
    private String scopeName;

    public String getScopeName() {
```

```
        String scopeName = "MyScope";
        ServiceContext sContext = ServiceContext.getContext();
        try {
            Scope scope =
                sContext.createScope(scopeName); // Creates a new scope
            this.scopeName = scope.getName(); // Returns the scope name
            sContext.setScope(scope); // Setting the Scope. This step is
important.
        } catch (ScopeAlreadyExistsException scopeExistsException) {
            System.out.println("Scope Already Exists...");
        }
        return this.scopeName;
    }


    public void setScopeName(String scopeName) {
        this.scopeName = scopeName;
    }

}
```

19.3.3 Setting an Outcome Parameter

The outcome parameter defines a Java method that is called after the page is created. In this method, you can add dynamic content to the page, refresh the user interface that calls up the dialog to confirm creation of a new page, or initiate browser navigation to the newly created page. To specify a Java method to invoke after page creation (that is, it is called after users click the **Create** button in the Create Page dialog), enter the following in the Edit Task Flow Binding dialog:

```
oracle_webcenter_page_createpage_outcome = ${'view.MyTestClass.go'}
```

You can define the method either with a String parameter or with no parameter. The String parameter value is the new page's name with full path. If the method is defined both ways, then the one with a parameter is invoked.

In the following example, the method `go(String newPagePath)` is invoked. This method redirects users to the newly created page with the Java class `MyTestClass.java`. The value of `newPagepath` is passed by the Page service.

```
package view;

import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;

public class MyTestClass {
    public MyTestClass() {
    }

    /*
       This method is executed only when there is no other go(String newPagePath)
method with a String argument .
    */
    public void go() {
        System.out.println("go() method without parameters executed.");
    }

    /*
       This method is prioritized and executed even when there are other go()
```

```
methods without arguments.
    */
    public void go(String newPagePath) {
        System.out.println("go(String newPagePath) method with parameter
executed.");
        try {
            FacesContext context = FacesContext.getCurrentInstance();
            String viewID = context.getViewRoot().getViewId();
            System.out.println("ViewID=" + viewID);

            ExternalContext extCtx = context.getExternalContext();
            String targetURI =
                extCtx.getRequestContextPath() + "/faces" + viewID;
            System.out.println("Navigate to: " + targetURI);
            extCtx.redirect(targetURI);
            context.responseComplete();
        } catch (Exception ee) {
            System.out.println("Error: " + ee.toString());
        }

    }
}
```

19.3.4 Specifying Styles

The Page service provides out-of-the-box styles that are packaged within the Page service libraries. You can use these styles or create your own custom styles.

This section describes how to apply out-of-the-box styles and custom styles. It contains the following subsections:

- Section 19.3.4.1, "Out-of-the-Box Styles"
- Section 19.3.4.2, "Custom Styles"

> **Note:** For more information about styles, see Section 19.6, "Introduction to Custom Styles and Templates."

19.3.4.1 Out-of-the-Box Styles

The following out-of-the-box-styles are provided (see Figure 19–1):

- Blank
- Left Narrow
- Right Narrow
- Three Column
- Stretch
- Text Page
- Web Page

To associate out-of-the-box styles with the Page - Create New task flow, use the optional parameter `oracle_webcenter_page_createpage_templatefile`. Use the parameter value to specify the location of the file that contains the list of out-of-the-box styles.

To associate out-of-the-box styles with the Page - Create New task flow:

1. Create a folder named `mystyles` under your Framework application's `public_html` folder.
2. Within `mystyles`, create a file named `default_pageservice_styles.xml`.
3. Add entries for all the out-of-the-box styles using the `templateDef` element.

   The `templateDef` element takes the following attributes:

   - `name`: This attribute specifies the path to the `.jspx` page that defines a page style. When you use the default styles, you must specify the path to the page that defines the style you intend to use. The default page style page files are located within the Page service libraries. For example:

     ```
     name="/oracle/webcenter/page/pstemplates/TemplateThreeColumn.jspx"
     ```

     `TemplateThreeColumn.jspx` is the page within the Page service library that defines the Three Column Template style.

   - `title`: The value you enter for `title` is used as the able for the style in the Create Page dialog. You can modify the title of an out-of-the-box style to any name. For example: `title="Three Column Layout"` or `title="Sales Standard"`.

   - `icon`: This attribute specifies an image icon for a style.

     For out-of-the-box styles, if you do not specify an icon for a style, then the Page service automatically uses the default icons assigned to each out-of-the-box style.

   Example 19–12 shows the content of a sample `default_pageservice_styles.xml` file that lists the other out-of-the-box styles:

Example 19–12 Sample default_pageservice_styles.xml File

```
<?xml version="1.0" encoding="UTF-8" ?>
 <templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateThreeColumn.jspx"
title="Three Column Layout" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateText.jspx"
title="Text Page" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateStretch.jspx"
title="Stretch" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateWeb.jspx"
title="Web Page" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateBlank.jspx"
title="Blank" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateNarrowLeft.jspx"
title="Left Narrow Column Layout" />
  <templateDef name="/oracle/webcenter/page/pstemplates/TemplateNarrowRight.jspx"
title="Right Narrow Column Layout" />
 </templatesDef>
```

4. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
5. Set the `oracle_webcenter_page_createpage_templatefile` parameter value to `${'/mystyles/default_pageservice_style.xml'}`.

   For example:

   ```
   oracle_webcenter_page_createpage_templatefile = ${'/mystyles/default_
   pageservice_style.xml'}
   ```

6. Click **OK**.

19.3.4.2 Custom Styles

In this example, you associate a custom style, labeled **News**, with the Page - Create New task flow.

To associate a custom style with the Page - Create New task flow:

1. Define your custom style in a `.jspx` file (for example, `NewsTemplate.jspx`).

   Note: For information about how to create ADF templates, see Section 19.6, "Introduction to Custom Styles and Templates."

2. Add an entry for the custom style in an XML file (for example, `custom_pageservice_style.xml`).

   For example:

```
<?xml version="1.0" encoding="UTF-8" ?>
  <templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
    <templateDef name="NewsTemplate.jspx" title="News" icon="/images/news.png"
/>
  </templatesDef>
```

   Note: For more information about the syntax to use for `templateDef` element and the style-related `.xml` file, see Section 19.3.4.1, "Out-of-the-Box Styles."

3. Save the `custom_pageservice_style.xml` under a folder in your application's `public_html` folder.

   For example:

```
…/public_html/mystyles/custom_pageservice_style.xml
```

4. The following files must be placed in the same folder as `custom_pageservice_style.xml`:

   - The file that defines the custom style, for example, `NewsTemplate.jspx`.
   - The page definition file of the custom style file, for example, `NewTemplatePageDef.xml`.
   - The ADF Template (for example, `defaultTemplate.jspx`) on which the custom style file is based. This step is applicable only if you are using ADF templates.

     Note: If you specify the location of an ADF template using the `oracle_webcenter_page_createpage_adftemplate` as described in following section, then you *do not* need to put this file (`defaultTemplate.jspx`) under the `mystyles` folder. You can specify a different location for the file by following Section 19.3.5, "Specify an ADF Template."

5. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."

6. Set the `oracle_webcenter_page_createpage_templatefile` parameter to `${'/mystyles/custom_pageservice_style.xml'}`, where `mystyles` is a folder under the `public_html` folder of your application.

   For example:

   ```
   oracle_webcenter_page_createpage_templatefile = ${'/mystyles/custom_
   pageservice_style.xml'}
   ```

19.3.5 Specify an ADF Template

When you want to use the same template for both design time and runtime pages, you can use an ADF template. All that is required is to point the Page - Create New task flow to the template location.

To associate an ADF template with the Page - Create New task flow:

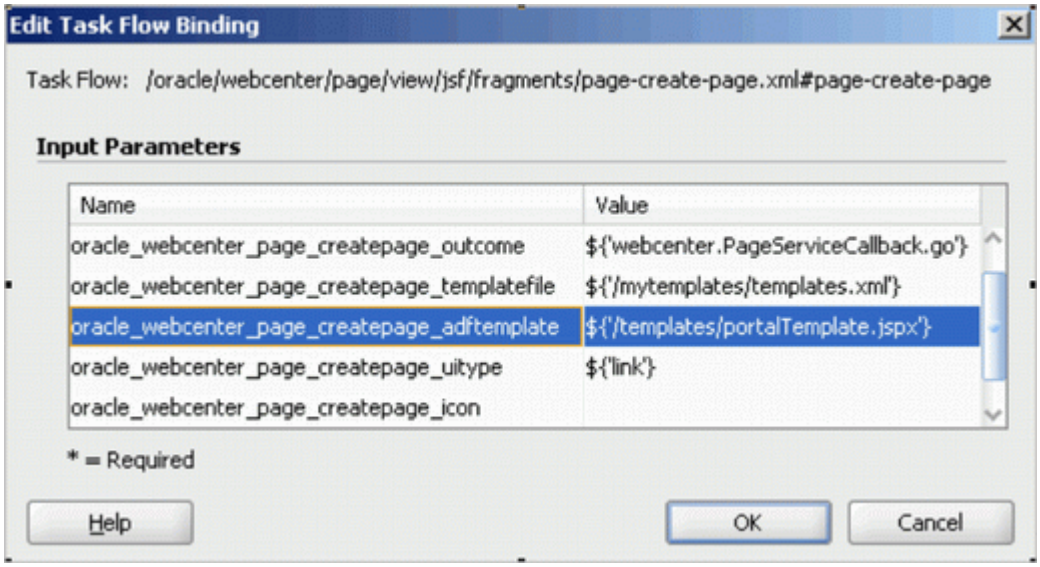
1. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
2. In the Value column next to the `oracle_webcenter_page_creatpage_adftemplate` parameter, enter the path to the ADF template.

   For example:

   ```
   ${'templates/portalTemplate.jspx'}
   ```

   In Figure 19–7, an ADF template called `portalTemplate.jspx` is used.

Figure 19–7 oracle_webcenter_page_createpage_adftemplate Parameter



ADF templates can be in your application's `public_html` folder. In this case, the path of `portalTemplate.jspx` is `../public_html/templates/portalTemplate.jspx`.

3. Click **OK**.

19.3.6 Showing a Command Link

By default, the Page - Create New task flow is rendered as a command *button*. You can instead specify that it is rendered as a command *link* (Figure 19–8).

Figure 19–8 Command Link



To render the Page - Create New task flow as a command link:

1. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
2. Revise the `oracle_webcenter_page_createpage_uitype` parameter to the following value: `${'link'}`.

   For example:

   ```
   oracle_webcenter_page_createpage_uitype = ${'link'}
   ```

3. Click **OK**.

19.3.7 Displaying an Image

You can associate an icon with the Page - Create New task flow button or link. For example, Figure 19–9 shows the task flow rendered as a command *link* with a page icon.

Figure 19–9 Command Link with a Page Icon



Figure 19–10 shows the task flow rendered as a command *button* with a page icon.

Figure 19–10 Command Button with a Page Icon



To specify an icon for the Page - Create New task flow button or link:

1. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
2. Revise the `oracle_webcenter_page_createpage_icon` parameter to the following value: `{'/images/page.jpg'}`.

   For example:

   ```
   oracle_webcenter_page_createpage_icon = ${'/images/page.jpg'}
   ```

3. Click **OK**.

19.3.8 Customizing the Label

By default, the label for the Page - Create New task flow command button or link is **Create Page**. You can provide your own value using the `oracle_webcenter_page_createpage_label` parameter value.

Figure 19–11 shows the task flow rendered as a command button with a custom label.

Figure 19–11 Custom Label



To change the command button or link label:

1. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
2. Revise the `oracle_webcenter_page_createpage_label` parameter to your preferred string value.

   For example:

   ```
   oracle_webcenter_page_createpage_label = ${'Create New Page'}
   ```

3. Click **OK**.

19.4 How to Create Task Flow View Pages

The following types of objects are required for the creation of a task flow view page:

- Task flow view page (JSFF file)
- Page definition of view page
- Task flow definition XML file, which defines the task flow

To enable the creation of task flow view pages through the Page - Create New task flow:

1. At design time, add a **Page - Create New** task flow to your page (`home.jspx`).
2. Open the Edit Task Flow Binding dialog as described in Section 19.3.1, "How to Access Page - Create New Task Flow Parameters."
3. Change the label on the Create Page button or link by revising the value specified for the `oracle_webcenter_page_createpage_label` parameter.

   For example:

   ```
   oracle_webcenter_page_createpage_label = ${'Create Task Flow'}
   ```

4. Click **OK**.
5. Create an XML file (for example, `custom_taskflow_styles.xml`) to contain entries for all the task flow styles.

   For example:

   ```
   <?xml version='1.0' encoding="UTF-8"?>
   <templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
     <templateDef name="mytaskflow_view.jsff" title="News"
   icon="/images/news.png" type="taskflow"/>
   </templatesDef>
   ```

   The element `templateDef` in the XML file contains the following attributes:

   - `name`: to specify the name of the task flow view page
   - `title`: to specify the label to associate with the task flow view page style in the Create Page dialog.
   - `icon`: to specify the image icon for the task flow

Note: For sample attribute syntaxes, see step 3 in Section 19.3.4.1, "Out-of-the-Box Styles."

6. Under your application's `public_html` folder, create a folder (for example, `mystyles`) and add the `custom_taskflow_styles.xml` file to it.
7. Make sure that the following files are also under the `mystyles` folder:
   - `mytaskflow_view.jsff` (task flow view page)
   - `mytaskflow_viewPageDef.xml` (task flow view page definition)
   - `mytaskflow_view.xml` (task flow definition)
8. Open the Edit Task Flow Binding dialog again, and provide the path to the XML file you created in step 5 to the `oracle_webcenter_page_createpage_templatefile` parameter.

   For example:

   ```
   oracle_webcenter_page_createpage_templatefile=${'/mystyles/custom_taskflow_
   styles.xml' }
   ```

   In this example, `mystyles` is a folder under the `public_html` folder of your application.
9. Run the `home.jspx` page, and then click the **Create Task Flow** button.
10. Enter the title of the task flow, select the **News** style (see Section 19.3.4.2, "Custom Styles"), and then click **Create**.

    The new task flow view page is rendered.

For information on where the task flows are created, see section Section 19.2.1.4, "Structure of Pages Created at Runtime."

19.5 Managing Pages

You can manage pages using either the Page service data control or the Page service APIs. This section contains the following subsections:

- Section 19.5.1, "Using the Page Service Data Control to Manage Pages"
- Section 19.5.2, "Using the Page Service APIs to Manage Pages and Task Flows"

19.5.1 Using the Page Service Data Control to Manage Pages

You can customize Page service views with the Page service data control. The data control enables you to view information about existing pages at runtime and delete any of the listed pages. The Page service data control, `PageServiceDC`, is included in the Page service libraries and is available at design time after you add the libraries to the application.

For more information about the Page service data control, see Section 19.8.2, "Using the Page Service Data Control."

19.5.2 Using the Page Service APIs to Manage Pages and Task Flows

The Page service provides APIs to manage pages and task flows. These include APIs to delete pages and task flows, change the hidden status, change the page scheme (name, background color, background image), and copy pages.

For more information about Page service APIs, see Section 19.8.1, "Using the Page Service Java APIs."

19.6 Introduction to Custom Styles and Templates

At runtime, when users click the Create Page option, the Create Page dialog displays a set of predefined styles for creating the page (see Figure 19–1). Users select one of these page styles to create a page based on the associated style template. The *style* provides a special look and feel to your pages. The *template* provides the page layout. Templates are the ADF pages on which runtime pages are based. Pages can be based on default or custom styles and templates.

To provide custom selections for the Create Page dialog, you must define them, for example, in `/mytemplates/templates.xml`. Place the directory `/mytemplates` under the web content root.

If you choose *not* to use the default options available with the Create Page dialog, then you can use the following types of custom templates in your application:

- Page and page fragment templates for use with runtime pages
- ADF templates for use as the base for page and page fragment templates
- Templates to be used by the Create Page dialog itself

This section explains how to customize the Page - Create New task flow to use different templates for the Create Page dialog and for the new pages that you create at runtime. It contains the following subsections:

- Section 19.6.1, "How to Create Templates for Pages Created at Runtime"
- Section 19.6.2, "Using ADF Templates"
- Section 19.6.3, "Creating Styles for the Create Page Dialog"

Note: For more information about the `templates.xml` file, see Section 19.3.4, "Specifying Styles."

19.6.1 How to Create Templates for Pages Created at Runtime

You can create custom page templates according to your design requirements and associate page styles with these templates. If you want to restrict the style options in the Create Page dialog or provide a different set of styles, then you must perform the following high-level tasks:

- Create pages or page fragments to be used as templates for the different page styles.
- Edit the Create Page dialog parameter to specify new page styles.

This section steps you through both procedures. It contains the following subsections:

- Section 19.6.1.1, "How to Create a Page or Page Fragment Style"
- Section 19.6.1.2, "How to Edit Create Page Dialog Styles"

19.6.1.1 How to Create a Page or Page Fragment Style

To create a page or page fragment style:

1. In Oracle JDeveloper, create a JSF template with all necessary information, such as header, welcome message, login/logout links, and so on.

2. When creating pages, create a page (JSPX) based on that template.

   When creating task flows, create a page fragment (JSFF) based on that template.

   When creating a page based on a template, the Page service actually copies the template and creates a new page based on it.

3. Put all the necessary content in the JSPX or JSFF file.

   For example, if there is a facet where you would like to display content, you can drop a Page Customizable component from Composer around it to make the page editable at runtime. Make sure you set the correct values for the `document title`, `styleClass`, and `inlineStyle` attributes.

   Note: If this page or page fragment will be used as a template for pages that can be customized at runtime, then you must ensure that the `ID` attribute on all components on this page or fragment is set to a unique value.

   Example 19–13 shows the sample code of a page fragment style, `pstemplateview.jsff`.

Example 19–13 Sample Code of a New Page Style

```
<?xml version='1.0' encoding='utf-8'?>
<!-- Copyright (c) 2006, 2008, Oracle and/or its affiliates.
All rights reserved. -->
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
     <jsp:directive.page contentType="text/html;charset=utf-8"/>
     <f:loadBundle basename="oracle.webcenter.page.view.resource.PGUIBundle"
                   var="res"/>
     <f:view>
       <af:document title="#{pageDocBean.title}" id="docrt">
         <af:form usesUpload="true" id="f1">
           <af:pageTemplate
viewId="/oracle/webcenter/page/pstemplates/defaultTemplate.jspx" id="T">
             <f:facet name="content">
               <pe:pageCustomizable id="pcl1">
                 <af:panelStretchLayout id="psl2"
                                        styleClass="replace_with_scheme_name"
                                        inlineStyle="replace_with_inline_style">
                   <f:facet name="center">
                     <af:panelGroupLayout id="pgl1"
                                          layout="scroll">
                       <pe:layoutCustomizable id="lc1"
                                 showLayoutChanger="#{pageServiceBean.isEditMode}"
                                 text="Change Layout"
                                 showIcon="false"
                                 type="threeColumnNarrow"
                                 shortDesc="Layout Changer">
                         <cust:panelCustomizable id="mainC"/>
                         <f:facet name="contentA">
                           <cust:panelCustomizable id="cnta"/>
                         </f:facet>
                         <f:facet name="contentB">
```

```
                        <cust:panelCustomizable id="cntb"/>
                      </f:facet>
                    </pe:layoutCustomizable>
                  </af:panelGroupLayout>
                </f:facet>
              </af:panelStretchLayout>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:pageTemplate>
      </af:form>
    </af:document>
```

Note: The page definition file of the JSPX page, which defines the style, must contain the following syntax for the `id` and `Package` attributes. The Page service replaces these values with the relevant values when creating a page based on a custom style:

```
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
                version="11.1.1.41.30"
                id="ps_pagedefusage"
                Package="ps_package">
```

Alternatively, the JSPX or JSFF template file that you are creating can be based on an ADF template.

Example 19–14 shows the sample code of an ADF template.

Note: If you choose to design your own ADF templates and want to set page style, then you must have a `Panel Group Layout` or `Panel Stretch Layout` component as a direct child of the `Page Customizable`. Additionally, you must set the placeholder for `styleClass` and `inlineStyle`, as shown in **bold** below. The page style values replace the placeholder text.

Example 19–14 Sample Code of a Page Style that uses an ADF Template

```
<?xml version='1.0' encoding='utf-8'?>
<!-- Copyright (c) 2006, 2008, Oracle and/or its affiliates.
All rights reserved. -->
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <jsp:directive.page deferredSyntaxAllowedAsLiteral="true"/>
  <jsp:directive.page contentType="text/html;charset=utf-8"/>
  <f:view>
    <af:document title="#{pageDocBean.title}" id="docrt">
      <af:form usesUpload="true" id="f1">
        <af:pageTemplate

viewId="/oracle/webcenter/webcenterapp/view/templates/WebCenterAppShellTemplate.js
px"
                value="#{bindings.shellTemplateBinding}" id="T">
```

```
          <f:facet name="content">
            <pe:pageCustomizable id="pcl1">
              <af:panelStretchLayout id="psl2"
                                     styleClass="replace_with_scheme_name"
                                     inlineStyle="replace_with_inline_style">
                <f:facet name="center">
                  <af:panelGroupLayout id="pgl1"
                                       layout="scroll">
                    <pe:layoutCustomizable id="lc1"

showLayoutChanger="#{pageServiceBean.isEditMode}"
                                           text="#{uib_o_w_w_r_WebCenter.LABEL_
CHANGE_LAYOUT}"
                                           showIcon="false" type="oneColumn"
                                           shortDesc="#{uib_o_w_w_r_
WebCenter.LABEL_CHANGE_LAYOUT}">
                      <cust:panelCustomizable id="mainC"/>
                      <f:facet name="contentA">
                        <cust:panelCustomizable id="cnta"/>
                      </f:facet>
                      <f:facet name="contentB">
                        <cust:panelCustomizable id="cntb"/>
                      </f:facet>
                    </pe:layoutCustomizable>
                  </af:panelGroupLayout>
                </f:facet>
              </af:panelStretchLayout>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:pageTemplate>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

Example 19–15 shows the sample code of a page style, `NewsTemplate.jspx`, that is based on an ADF template, `portalTemplate.jspx`. The `viewId` attribute of the `af:pageTemplate` tag provides the name of the ADF template used.

For information about using ADF templates, see the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

Example 19–15 Sample Code of a Page Style on an ADF Template

```
<?xml version='1.0' encoding='utf-8'?>
<!-- Copyright (c) 2006, 2008, Oracle and/or its affiliates.
All rights reserved. -->
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <jsp:directive.page contentType="text/html;charset=utf-8"/>
  <f:loadBundle basename="oracle.webcenter.page.view.resource.PGUIBundle"
                var="res"/>
  <f:view>
    <af:document title="#{pageDocBean.title}">
```

```
      <af:form usesUpload="true">
          <af:pageTemplate viewId="/mytemplates/portalTemplate.jspx"
 value="#{bindings.pageTemplateBinding}">
      <f:facet name="pageContent">
            <pe:pageCustomizable id="pgc1">
              <af:panelStretchLayout id="psl1"
                                     styleClass="#{pageDocBean.CSSStyle}"
                                     inlineStyle="#{pageDocBean.inlineStyle}">
                <f:facet name="center">
                  <af:panelGroupLayout id="pgl1" layout="scroll">
                    <pe:layoutCustomizable id="lc1"
                        showLayoutChanger="#{pageServiceBean.isEditMode}"
                                          text="#{res['TEMPLATE.CHANGE_LAYOUT']}"
                                          showIcon="false" type="oneColumn"
                                          shortDesc="#{res['TEMPLATE.CHANGE_
LAYOUT']}">
                      <cust:panelCustomizable id="mainC">
                        <cust:showDetailFrame id="sdf2"
                                              showMinimizeAction="none"
                                              showMoveAction="none"
                                              showRemoveAction="none"
                                              displayHeader="false"
                                              displayShadow="false"
                                              stretchContent="false"
                                              shortDesc="#{null}"
                                              background="light"
                                              showResizer="never"
                                              text="Text">
                          <af:richTextEditor id="rtepc" clientComponent="true"
                                             simple="true" label="#{null}"
                                             rows="20" readOnly="false"
                                             contentStyle="width:100%;"/>
                        </cust:showDetailFrame>
                      </cust:panelCustomizable>
                      <f:facet name="contentA">
                        <cust:panelCustomizable id="cnta"/>
                      </f:facet>
                      <f:facet name="contentB">
                        <cust:panelCustomizable id="cntb"/>
                      </f:facet>
                    </pe:layoutCustomizable>
                  </af:panelGroupLayout>
                </f:facet>
              </af:panelStretchLayout>
              <f:facet name="editor">
                <pe:pageEditorPanel/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:pageTemplate>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

To display a different style, or use a different template for a style, you must define it, for example, in `/mytemplates/templates.xml`. Place the directory `/mytemplates` under the web content root. For more information about the `templates.xml` file, see Section 19.3.4, "Specifying Styles."

Example 19–16 shows a sample `templates.xml` file.

Example 19–16 templates.xml File

```
<?xml version='1.0' encoding="UTF-8"?>
<templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
  <templateDef name="TemplateBlank.jspx"
               title="Blank"/>
  <templateDef name="TemplateNarrowLeft.jspx"
               title="Left Narrow Column Layout"/>
  <templateDef name="TemplateNarrowRight.jspx"
               title="Right Narrow Column Layout"/>
  <templateDef name="TemplateThreeColumn.jspx"
               title="Three Column Layout"/>
  <templateDef name="TemplateStretch.jspx"
               title="Stretch"/>
  <templateDef name="TemplateText.jspx"
               title="Text"/>
  <templateDef name="TemplateWeb.jspx"
               title="Web"/>
  <templateDef name="NewsTemplate.jspx"
               title="News"/>
</templatesDef>
```

19.6.1.2 How to Edit Create Page Dialog Styles

To edit the styles for the Create Page dialog and add a reference to the new page style:

1. Create the `/mytemplates/templates.xml` file, commenting out `templateDef` entries for any default page styles that you do not want to display.
2. In the `templateDef` entry for the style that you want to associate with a different template, edit the `name` and `type` attributes, as shown in the following example:

   ```
   <templateDef name="pstemplateview.jsff"
                title="News"
                icon="/images/news.png"
                type="taskflow"/>
   ```

   where

   - `name` is the name of the page style that you want to use.
   - `title` is the label displayed for the style in the Create Page dialog.
   - `type` is the type of page style. It can take the value `page` or `task flow`. The default value `page`, creates a page based on the style's associated template. The value `task flow` creates a task flow view of a page based on the style's associated template.
3. If you want to add a new style to the Create Page dialog, add a `templateDef` entry, as shown in the following example:

   ```
   <templateDef name="pstemplateview3.jsff"
                title="Rich Text"
                type="taskflow"
                forGroupSpace="false"/>
   ```

4. Save the XML file.

At runtime, the Create Page dialog displays the styles you specified. The pages and task flow page views users create using those styles are based on the JSPX or JSFF templates you defined.

19.6.2 Using ADF Templates

If your page template is based on an ADF template, you can configure the page creation task flow to use a different ADF template depending on different criteria, such as user or scope.

You can specify the ADF template to use for runtime page creation in either of the following ways:

- By using the `oracle_webcenter_page_createpage_adftemplate` parameter in the page creation task flow (see Section 19.3.4, "Specifying Styles").
- By using the `ADFTemplateViewID` parameter in the `createPage()` and `createTaskflow()` APIs.

Any JSPX page template that is based on an ADF template contains an `af:pageTemplate` tag with the `viewId` attribute. The `viewId` attribute contains the name of the ADF template. Example 19–15 shows a page template that is based on the ADF template `portalTemplate.jspx`.

When you specify an ADF template name for use during page creation, the Page service searches for the `af:pageTemplate` tag in the page template and updates the `viewId` attribute with the value you provided.

This section describes how to specify which ADF template to use under a given circumstance. It contains the following subsections:

- Section 19.6.2.1, "How to Specify the ADF Template Name Using the Task Flow Parameter"
- Section 19.6.2.2, "How to Specify the ADF Template Name Using API Parameters"

19.6.2.1 How to Specify the ADF Template Name Using the Task Flow Parameter

To specify the ADF template name using the task flow parameter:

1. Open the JSPX file containing the Page - Create New task flow.
2. Right-click the page name and select **Go to Page Definition**.
3. In the page definition file, under Executables, select the Create Page task flow, and click the **Edit** icon on the header.
4. The Edit Task Flow Binding dialog opens with the list of input parameters supported by the task flow.
5. Set the `oracle_webcenter_page_createpage_adftemplate` parameter.

   If you define a constant value, for example `${'/mytemplates/defaultTemplate2.jspx'}`, then this ADF template is used for all pages created using the create page task flow.

   However, by providing an EL value for the parameter, you can ensure that a different ADF template is used based on different criteria, such as user or scope.

   For example, assume you have two users, `user1` and `user2`, and want to use a different ADF template for pages created by each user. You must enter the value `${MyClass.ADFTemplate}` for the `oracle_webcenter_page_createpage_adftemplate` parameter and define the method `MyClass.getADFTemplate()` so that it returns different ADF templates based on which user has logged in.

6. Click **OK**.

19.6.2.2 How to Specify the ADF Template Name Using API Parameters

To specify the ADF template name using API parameters:

1. Create a page or a task flow, and specify the ID of the ADF template view activity.

   The page or task flow is subsequently created based on that ADF template.

   To create a page:

   ```
   PageService.createPage(
        String pageType, String nameFormat, String title,
        String pageTemplate, String pageTemplatePath,
        String ADFTemplateViewID,
        String cssStyle, String schemeBGImage, String schemeBGColor)
   ```

   where `ADFTemplateViewID` is the ID of the ADF template view activity.

   To create a task flow:

   ```
   PageService.createTaskflow(
        String nameFormat, String title,
        String pageTemplate, String pageTemplatePath,
        String ADFTemplateViewID,
        String cssStyle, String schemeBGImage, String schemeBGColor)
   ```

   where `ADFTemplateViewID` is the ID of the ADF page template view activity.

19.6.3 Creating Styles for the Create Page Dialog

To use your own page styles, define them in `/mytemplates.templates.xml` file. If you want to maintain a separate XML file with a different set of page styles, you can create an XML file and reference that from the Page - Create New task flow. For example, to display two different options to two users with different privileges, then you must create two XML files and define different styles in each XML file. You can then ensure that the Create Page dialog displays the respective options to each user.

This section describes how to create and reference a new style for the runtime Create Page dialog. It contains the following subsections:

- Section 19.6.3.1, "How to Create a Style for the Create Page Dialog"
- Section 19.6.3.2, "How to Reference a New Style from the Page - Create New Task Flow"

19.6.3.1 How to Create a Style for the Create Page Dialog

To create a style for the Create Page dialog:

1. In the `/mytemplates` directory, create an XML file, for example `templates2.xml`.
2. Add a `templatesDef` element, and within that add a `templateDef` entry for each style that you want to include in the Create Page dialog.

   Example 19–17 shows the code entered into sample XML file `templates2.xml` that references the `pstemplateview.jsff` fragment you created in the previous section.

Example 19–17 Sample Code for the Create Page Dialog Style

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
- <templatesDef xmlns="http://xmlns.oracle.com/webcenter/page">
<templateDef name="pstemplateview.jsff" title="News" icon="/images/news.png"
             type="taskflow" />
  </templatesDef>
```

where

- `name` is the name of the page or page fragment to use as a template.
- `title` is the label for the style in the Create Page dialog.
- `type` is the type of template. It can take that value `page` or `taskflow`. The default value `page` creates a page based on the style's associated template. The value `taskflow` creates a task flow view of a page based on the style's associated template.

3. Save the XML file.

19.6.3.2 How to Reference a New Style from the Page - Create New Task Flow

To reference the new XML file from the Page - Create New task flow:

1. Open the JSPX page that contains the Page - Create New task flow.
2. Right-click the JSPX page, and select **Go to Page Definition**.
3. In the page definition file, under Executables, select the Create Page task flow, and click the **Edit** icon on the header.

   The Edit Task Flow Binding dialog opens with the list of input parameters supported by the task flow.
4. Provide the template file name for the `oracle_webcenter_page_createpage_templatefile` parameter.

   If you define a constant value, for example `${'/mytemplates/templates2.xml'}`, then this template is used for the Create Page dialog always.

   However, by providing an EL value for the parameter, you can ensure that a different template is used for the dialog based on different criteria, such as user or scope.

   For example, assume you have two users, `user1` and `user2`, and want to display a different page style to each user. In this case, you must have two templates for the Create page dialog. Let us say you have created a new template, `templates2.xml`, and updated the default template, `templates.xml`. You can then specify `${TemplateBean.template}` for the `oracle_webcenter_page_createpage_templatefile` parameter.

   To use this value, you must first create the managed bean, `TemplateBean.java`, with method `getTemplate()`, which returns `templates.xml` for `user1` and `templates2.xml` for `user2`. At runtime, user1 and user2 are each shown different options in the Create Page dialog, as shown in Figure 19–12 and Figure 19–13.

Figure 19–12 Create Page Dialog using the templates.xml Style

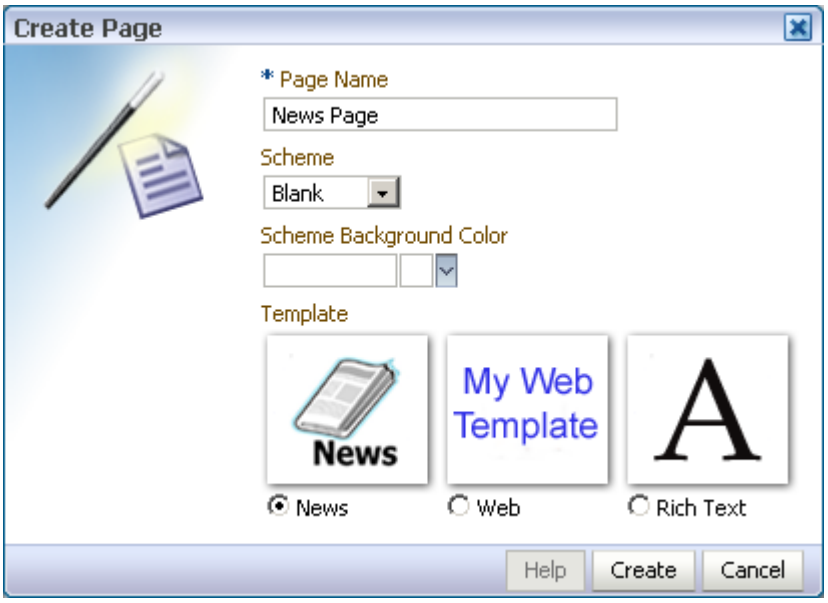
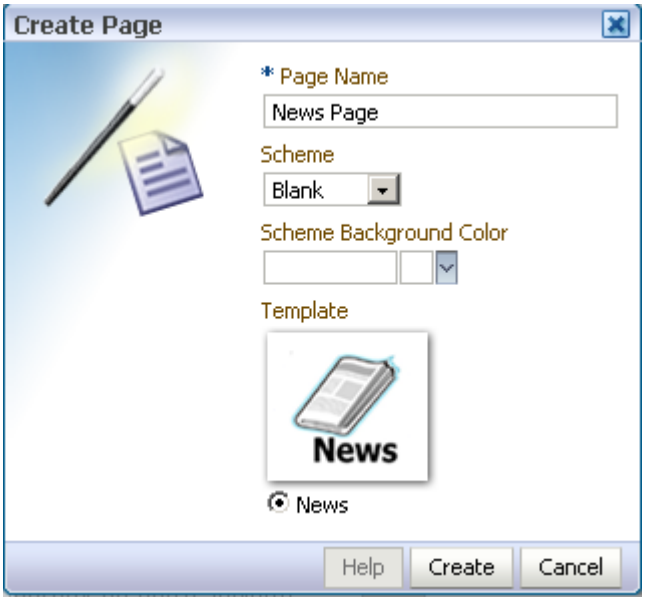


Figure 19–13 Create Page Dialog using the templates2.xml Style



5. Click **OK**.
6. Click **Save All** to save your work.

19.7 Customizing Page Service Views

This section describes various ways to customize Page service views. It contains the following subsections:

- Section 19.7.1, "Rendering Pages with ADF Faces Components"
- Section 19.7.2, "Managing User Security on Pages and Task Flows"

19.7.1 Rendering Pages with ADF Faces Components

You can use ADF Faces components to render runtime pages. For example, you can use ADF Faces components to render pages as tabs, links, and image links—or thumbnail views of pages. This section provides some examples of the types of pages you can create using ADF Faces components. It contains the following subsections:

- Section 19.7.1.1, "Rendering Pages as Tabs Using ADF Faces Components"
- Section 19.7.1.2, "Rendering Pages as Links Using ADF Faces Components"
- Section 19.7.1.3, "Rendering Pages as Image Links Using ADF Faces Components"

19.7.1.1 Rendering Pages as Tabs Using ADF Faces Components

When you design your application to render pages as tabs, you can use a range of ADF Faces components to accomplish this. This section provides information about the ADF Faces components that support rendering pages as tabs. It contains the following subsections:

- Section 19.7.1.1.1, "Rendering Pages as Tabs Using af:navigationPane"
- Section 19.7.1.1.2, "Rendering Pages as Tabs Using af:panelTabbed"

19.7.1.1.1 Rendering Pages as Tabs Using af:navigationPane Example 19–18 illustrates how pages can be rendered as tabs using `navigationPane` components.

Here a method called `pages` from a custom bean (`MyPageServiceBean`) is used to get the list of pages. For more information about getting the list of pages with the Page service API, see the `getPages()` method in Page service API section.

Example 19–18 Rendering Pages as Tabs Using af:navigationPane

```
<af:navigationPane id="tabs">
  <af:forEach var="tab" items="#{MyPageServiceBean.pages}">
    <af:commandNavigationItem text="#{tab.title}"  id="cni1"/>
    </af:forEach>
  </af:navigationPane>
```

The output of the code depicted in Figure 19–18 renders as a series of tabs (Figure 19–14).

Figure 19–14 Render Pages as Tabs



19.7.1.1.2 Rendering Pages as Tabs Using af:panelTabbed Example 19–19 illustrates how pages can be rendered as tabs using `navigationPane` components.

Here a method called `pages` from a custom bean (`MyPageServiceBean`) is used to get the list of pages. For more information about getting list of pages using Page service API, see the `getPages()` method in Page service API section.

Example 19–19 Rendering Pages as Tabs Using af:panelTabbed

```
<af:panelTabbed id="pt1" >
    <af:forEach var="tab" items="#{ MyPageServiceBean.pages}">
        <af:showDetailItem text="#{tab.title}" id="sdi1"/>
     </af:forEach>
   </af:panelTabbed>
```

The output of the code in Example 19–19 renders as a series of tabs (Figure 19–15).

Figure 19–15 Render Pages as af:panelTabbed Component



19.7.1.2 Rendering Pages as Links Using ADF Faces Components

This section provides examples of using ADF Faces Components to render pages as different types of links. It includes the following subsections:

- Section 19.7.1.2.1, "Rendering Pages as Links Using af:commandLink"
- Section 19.7.1.2.2, "Rendering Pages as Links Using af:goLink"

19.7.1.2.1 Rendering Pages as Links Using af:commandLink Example 19–20 illustrates how pages can be rendered as links using `af:commandLink`.

Example 19–20 Rendering Pages as Links Using af:commandLink

```
<af:panelGroupLayout id="pgl1" layout="vertical">
  <af:forEach var="tab" items="#{MyPageServiceBean.pages}">
     <af:commandLink text="#{tab.title}" id="cl1"/>
   </af:forEach>
</af:panelGroupLayout>
```

The output of the code depicted in Example 19–20 renders as a list of links (Figure 19–16).

Figure 19–16 Render Pages as Links



19.7.1.2.2 Rendering Pages as Links Using af:goLink Example 19–21 illustrates how pages can be rendered as links using `af:goLink`.

Example 19–21 Rendering Pages as Links Using af:goLink

```
<af:panelGroupLayout id="pgl2" layout="vertical">
     <af:activeOutputText value="List Of Pages" id="aot2"
                          inlineStyle="font-weight:bold; font-size:small;"/>
      <af:forEach var="tab" items="#{pageServiceBean.pages}">
```

```
        <af:goLink text="#{tab.title}" id="gl1"/>
      </af:forEach>
    </af:panelGroupLayout>
```

The output of the code depicted in Example 19–21 renders as a list of links (Figure 19–16).

19.7.1.3 Rendering Pages as Image Links Using ADF Faces Components

Image links are linked thumbnail views of a page that users click to access the full page view. Example 19–22 illustrates how pages can be rendered as image links.

Example 19–22 Rendering Thumbnail Views of Pages

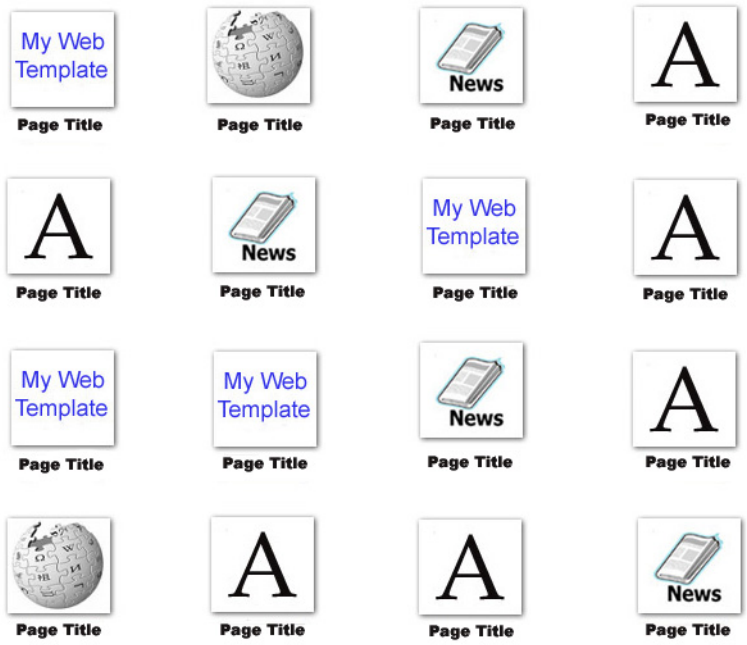
```
  <af:panelGroupLayout id="pg1_11" layout="scroll">
          <af:panelGroupLayout id="pg11" layout="horizontal" halign="center"
                               valign="middle">
            <h:panelGrid id="ds" columns="4" cellpadding="10" cellspacing="5"
                         style="text-align: center;" rowClasses="bottomAlign">
              <af:forEach varStatus="stat" begin="0"
end="#{MyPageServiceBean.pageCount}">
                <af:panelGroupLayout id="sdsd" layout="vertical"
                                     halign="center">
                  <af:commandImageLink text="" id="cil1"
                                       icon="/<image_name>}"/>
                  <af:activeOutputText value="Page Title" id="aot3"/>
                </af:panelGroupLayout>
              </af:forEach>
            </h:panelGrid>
          </af:panelGroupLayout>
        </af:panelGroupLayout>
```

In Example 19–22, `pageCount` is a method from a custom bean (`MyPageServiceBean`) that gets the number of pages in the current scope. Users must write this method on their own using the other methods from Page service APIs.

In Example 19–22, `<image_name>` is the name of the image with which you associate your page.

The output of the code in Example 19–22 is rendered as a linked series of thumbnail views of a page (Figure 19–17).

Figure 19–17 Render Pages as Image Links



19.7.2 Managing User Security on Pages and Task Flows

You can manage access to pages and task flows either through the Security Panel, which is a part of Composer, or through Page service APIs.

For more information about runtime security, see the "Setting Page Access" section in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

For more information about setting page access with the Page service APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

19.8 Advanced Information for the Page Service

This section describes the APIs and data controls available for the Page service. It includes the following sections:

- Section 19.8.1, "Using the Page Service Java APIs"
- Section 19.8.2, "Using the Page Service Data Control"

19.8.1 Using the Page Service Java APIs

This section provides information about Page service Java APIs. It includes information about the location of these APIs, how to make your application ready to use them, and how to use them to create pages. It contains the following subsections:

- Section 19.8.1.1, "Configuration Settings Required for Using Page Service APIs"
- Section 19.8.1.2, "Introduction to the Page Service APIs"
- Section 19.8.1.3, "How to Set up Your Application to Use the Page Service APIs"
- Section 19.8.1.4, "Example: How to Create a Page"

19.8.1.1 Configuration Settings Required for Using Page Service APIs

To use the Page service APIs, you must perform the following steps:

1. Confirm that you have the following entries in your adf-config.xml file:

```
<namespace path="/oracle/webcenter/page/scopedMD"
metadata-store-usage="WebCenterFileMetadataStore"/>
<namespace path="/pageDefs"
metadata-store-usage="WebCenterFileMetadataStore"/>
<namespace path="/mytemplates"
metadata-store-usage="WebCenterFileMetadataStore"/>
```

   If these entries do not exist in adf-config.xml, add them using a text editor.

2. Configure the .cpx file to use dynamic page mapping:

```
<Application xmlns="http://xmlns.oracle.com/adfm/application"
             version="11.1.1.51.35" id="DataBindings" SeparateXMLFiles="false"
             Package="view" ClientType="Generic"
             PageMapClass="oracle.jbo.uicli.mom.DynamicPageMapImpl"
             BasePageDefPackageName="pageDefs">
  <definitionFactories>
    <factory nameSpace="http://xmlns.oracle.com/adf/controller/binding"

className="oracle.adf.controller.internal.binding.TaskFlowBindingDefFactoryImpl
"/>
    <dtfactory
className="oracle.adf.controller.internal.dtrt.binding.BindingDTObjectFactory"/
>
  </definitionFactories>
  <pageMap/>
    <page path="/page1.jspx" usageId="view_page1PageDef"/>
  </pageMap>
  <pageDefinitionUsages/>
    <page id="view_page1PageDef" path="view.pageDefs.page1PageDef"/>
  </pageDefinitionUsages>
  <dataControlUsages>
    <dc id="PageServiceDC"
        path="oracle.webcenter.page.internal.model.PageServiceDC"/>
  </dataControlUsages>
</Application>
```

   If these entries do not exist in your .cpx file, add them using a text editor.

19.8.1.2 Introduction to the Page Service APIs

The Page service APIs are located in the `oracle.webcenter.page.model.PageService` class.

The Page service provides the following APIs:

- createPage - see Section 19.8.1.4, "Example: How to Create a Page"
- deletePage
- createTaskflow - see Example 19–23, "Using Page Service APIs to Create a Task Flow View Page"
- deleteTaskflow
- createScope
- deleteScope

- copyPage
- getPageList - see Example 19–24, "Using Page Service APIs to Get a List of Pages"
- setPageTitle - see Example 19–25, "Using Page Service APIs to Set the Page Title"
- changeHiddenStatus
- changePageScheme - see Example 19–26, "Using Page Service APIs to Set the Page Scheme"
- setPageAccess

For more information about the Page service APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

Example 19–23 Using Page Service APIs to Create a Task Flow View Page

```
PageDef createTaskflow(String nameFormat, String title, String pageTemplate,
  String pageTemplatePath, String cssStyle, String schemeBGImage, String
schemeBGColor)
  throws DuplicateNameException, InvalidNameException, LockUnavailableException;
```

where

- `nameFormat` is the name format of the task flow; it can contain a subpath, such as `users/UserA/taskflow{0}.jspx`.
- `title` is the title of the task flow.
- `pageTemplate` is the template used for the task flow.
- `pageTemplatePath` is the path for the task flow template.
- `cssStyle` is the cascading style sheet file for this task flow.
- `schemeBGImage` is the background image for the custom scheme.
- `schemeBGColor` is the background color for the custom scheme.

It returns the `PageDef` of the newly created task flow and throws the following:

- `InvalidNameException` if the name supplied cannot be used for update.
- `DuplicateNameException` if the page with the same name already exists.
- `LockUnavailableException` if another page or task flow operation is in progress.

Example 19–24 Using Page Service APIs to Get a List of Pages

```
PageDef getPageList(String path)
```

where

- `path` the path under which you want to see a list of all page that the user can view. If you do not specify a path, it returns a list of all pages that the user can view in the current scope.

It returns the list of all pages that are not hidden in the current scope or under a specific path.

Example 19–25 Using Page Service APIs to Set the Page Title

```
setPageTitle(String pagepath, String title)
  throws InvalidNameException, ObjectChangedException, LockUnavailableException,
oracle.webcenter.framework.translations.exception.TranslationsException;
```

where

- `pagepath` is the path of the page for which you want to change the title.
- `title` is the new name of the page.

It updates the name of the page and throws the following:

- `InvalidNameException` if the name supplied cannot be used for update.
- `ObjectChangedException` if the page has changed since it was last retrieved.
- `LockUnavailableException` if another page operation is in progress.

Example 19–26 Using Page Service APIs to Set the Page Scheme

```
setPageScheme(String pagepath, String schemeName, String schemeBGImage, String
schemeBGColor, String otherCSS)
```

where

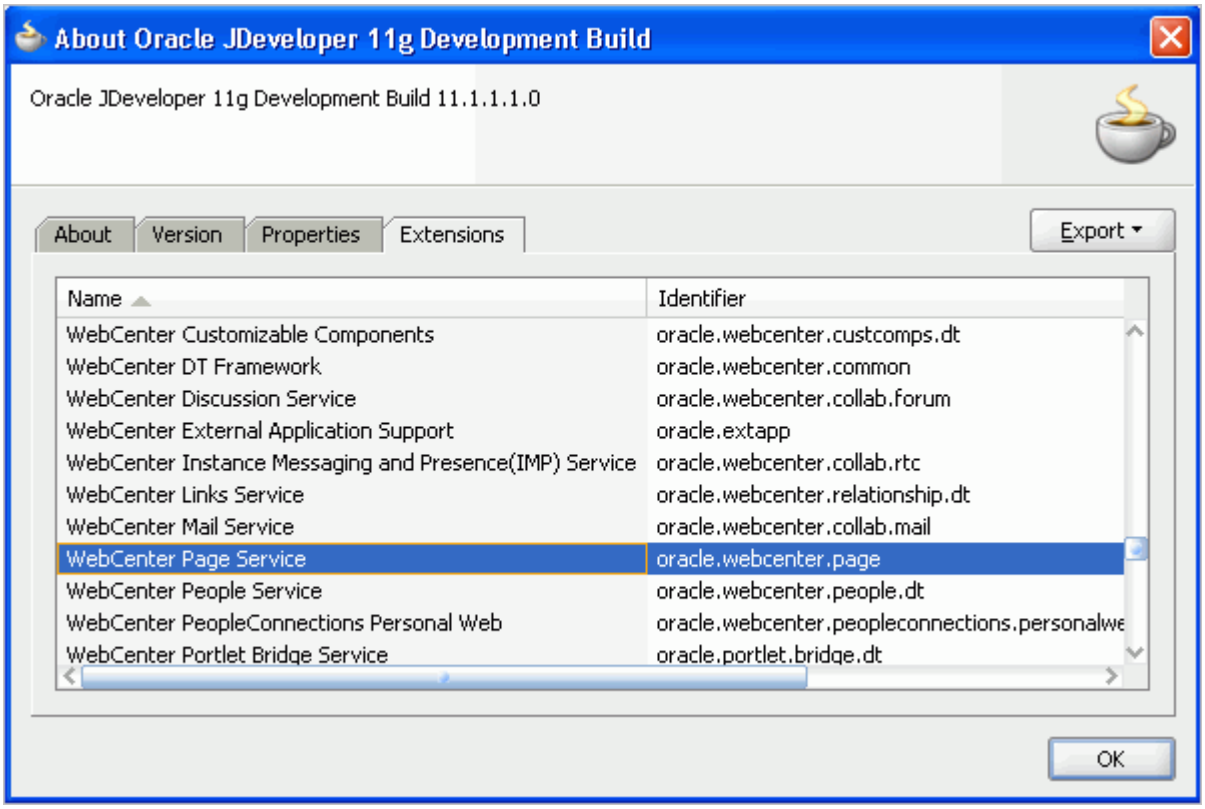
- `pagepath` the path of the page for which you want to change the scheme.
- `schemeName` the name of the CSS for the page.
- `schemeBGImage` the background image for the custom scheme.
- `schemeBGColor` the background color for the custom scheme.
- `otherCSS` any other CSS attributes you want to specify.

It updates the page's scheme properties in the jspx page.

19.8.1.3 How to Set up Your Application to Use the Page Service APIs

To use the Page service APIs, WebCenter Portal's Page service extension must be present in your JDeveloper application. In JDeveloper, go to the **Help - About** menu, and click the **Extensions** tab. Look for the WebCenter Page Service, as shown in Figure 19–18.

Figure 19–18 JDeveloper Extensions



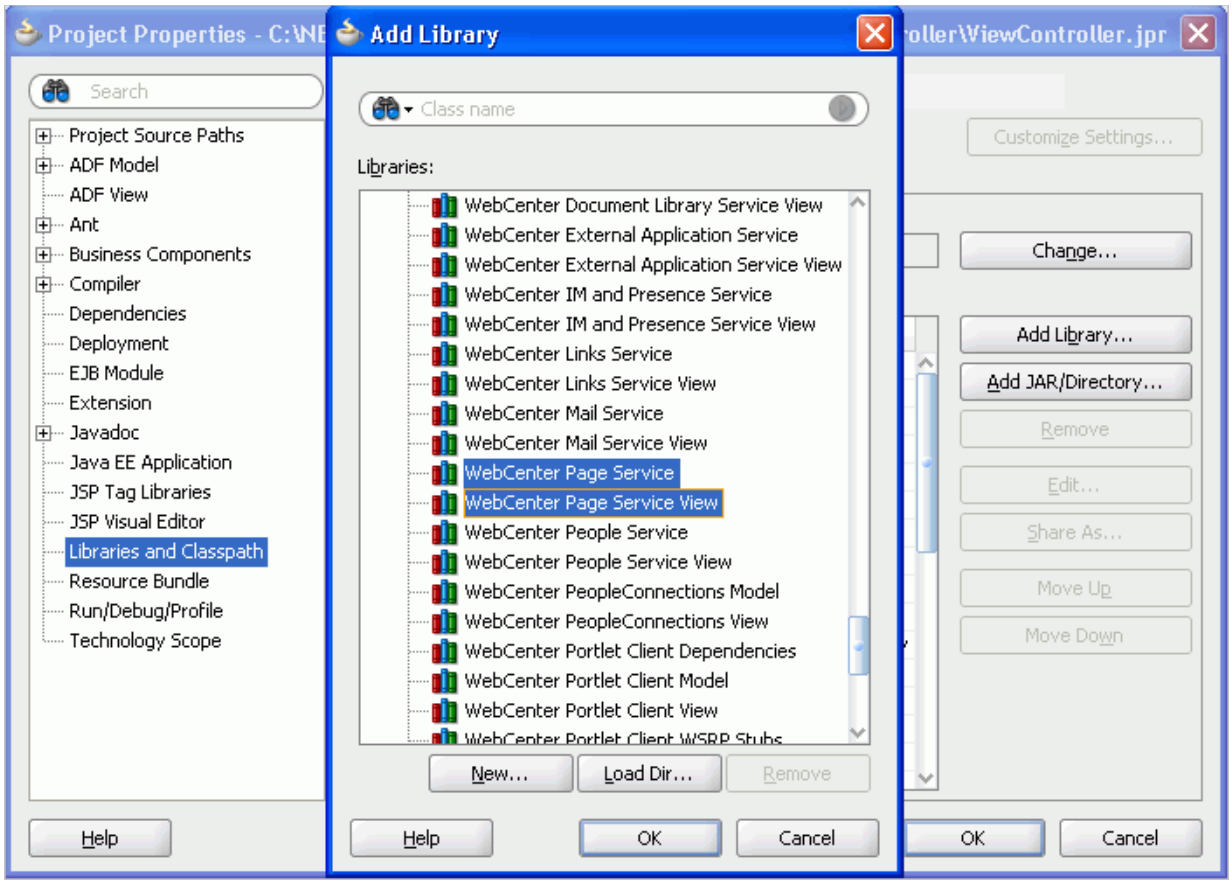
Note: For information about adding the WebCenter Portal's extensions, see Chapter 3, "Preparing Your Development Environment."

After confirming that the Page service is included with the WebCenter Portal's extensions, you must add the Page service libraries to the project.

To add Page service libraries to your project:

1. Right-click your **ViewController** project and select **Project Properties** and then **Libraries and Classpath**.
2. Click **Add Library** and select the **WebCenter Page Service** and **WebCenter Page Service View** libraries, as shown in Figure 19–19.

Figure 19–19 Adding Page Service Libraries



3. Click **OK**.

If you do not want to use JDeveloper as the code editor, then you can set the CLASSPATH to the following jar files:

- `Jdev_Home/jdeveloper/webcenter/jlib/pagem.jar`
- `Jdev_Home/jdeveloper/webcenter/jlib/pages.jar`
- `Jdev_Home/jdeveloper/webcenter/jlib/page-service-view.jar`
- `Jdev_Home/jdeveloper/webcenter/jlib/page-service-skin.jar`

19.8.1.4 Example: How to Create a Page

This section provides an example of how to create a page using the Page service APIs. Your requirements may call for different methods. Use the way the `createPage` method is invoked in this sample code as a model for invoking the methods you require on the `PageService` object.

Typically, pages created using Page service APIs are based on templates. Before you start creating pages using the APIs, you must ensure that each template file, for example `MyPageTemplate1.jspx`, has a page definition file, in this case `MyPageTemplate1PageDef.xml`, in the same directory as the JSPX page.

Example 19–27 shows how to use the `createPage() API` to create a scope named `MyScope`, create an MDS session instance, instantiate the Page service class with this new scope, and then create a page within this scope. The new page is created based on the `MyPageTemplate1.jspx` template.

Example 19–27 Sample Showing How to Create a Page

```
...

public void createMyPage
{
    try
    {
      MDSSession mdsSess =
(MDSSession)ADFContext.getCurrent().getMDSSessionAsObject();
      PageServiceConfig config = new PageServiceConfig(mdsSess, "defaultScope");
      mPageService = PageServiceFactory.createInstance(config);

      String pageNameFormat ="Page{0}.jspx";
      String pageTitle ="New Page";

      //
      // The template used to create the new page is
/mytemplates/MyPageTemplate1.jspx
      // path "/mytemplates" should be defined as a MDS namespace.
      // Also MyPageTemplate1.jspx should have a pageDef called
"MyPageTemplate1PageDef.xml and
      // it has the same path as the jspx file.

      PageDef newPage = mPageService.createPage(
        "personaluserpage", pageNameFormat, pageTitle,
        "MyPageTemplate1.jspx",
        "/mytemplates/",
        null, null, null);

    }
    catch(Exception e)
    {
      // Handle exception
    }
}

...
```

> **Note:** Pages and task flows created using Page service APIs do not have permissions granted by default. You must explicitly define permissions on the page.

For more information about the Page service APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

19.8.2 Using the Page Service Data Control

The Page service data control enables you to view information about existing pages at runtime and delete any of the listed pages.

This section describes how to add a Page service data control to a project and how to use it to view, edit, and delete pages. It contains the following subsections:

- Section 19.8.2.1, "How to Add the Page Service Data Control"
- Section 19.8.2.2, "How to View, Edit, and Delete Pages at Runtime"

19.8.2.1 How to Add the Page Service Data Control

This section provides an example of adding the Page service data control at design time.

To add a Page service data control to your project:

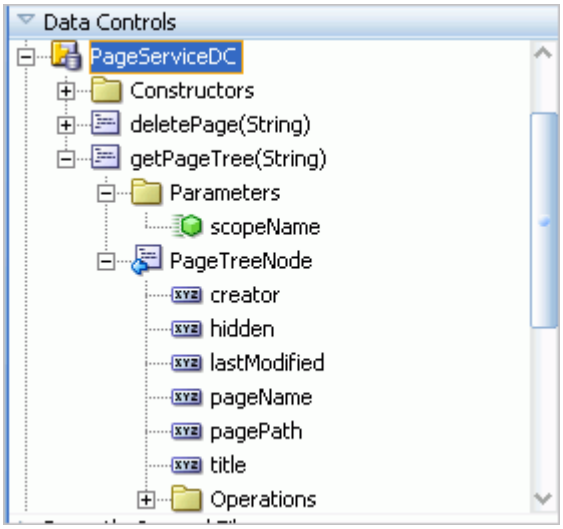
1. Ensure that you can see the PageServiceDC in the Application Navigator.

   The PageServiceDC appears in the Application Navigator after you have performed either of the following actions:

   - Added the **Page - Create New** task flow to your project (see Section 19.2.1.1, "How to Add the Page - Create New Task Flow").
   - Added the **Page Data Control** as described in Section 7.1.3, "Using WebCenter Portal Data Controls.".

2. In the Data Controls panel of the Application Navigator, expand **PageServiceDC**, as shown in Figure 19–20.

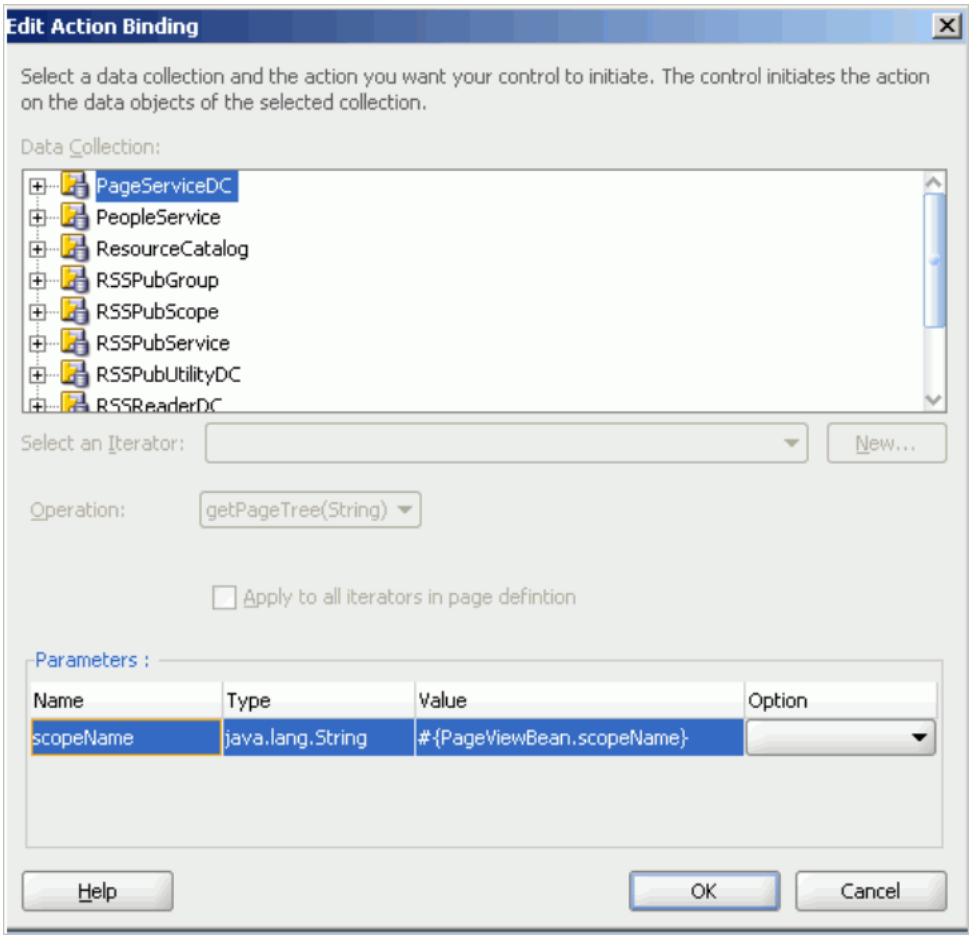
Figure 19–20 Page Service Data Control



3. Under **getPageTree()**, drag **PageTreeNode** and drop it on the page.
4. From the **Create** menu, select **Table**, and then select **ADF Read-only Table**.
5. In the dialog, enter the `scopeName` parameter to view all the pages listed under the specified scope.

   If you do not enter any value, then the data control takes the default scope. In Figure 19–21, the scope value is fetched from a custom managed bean.

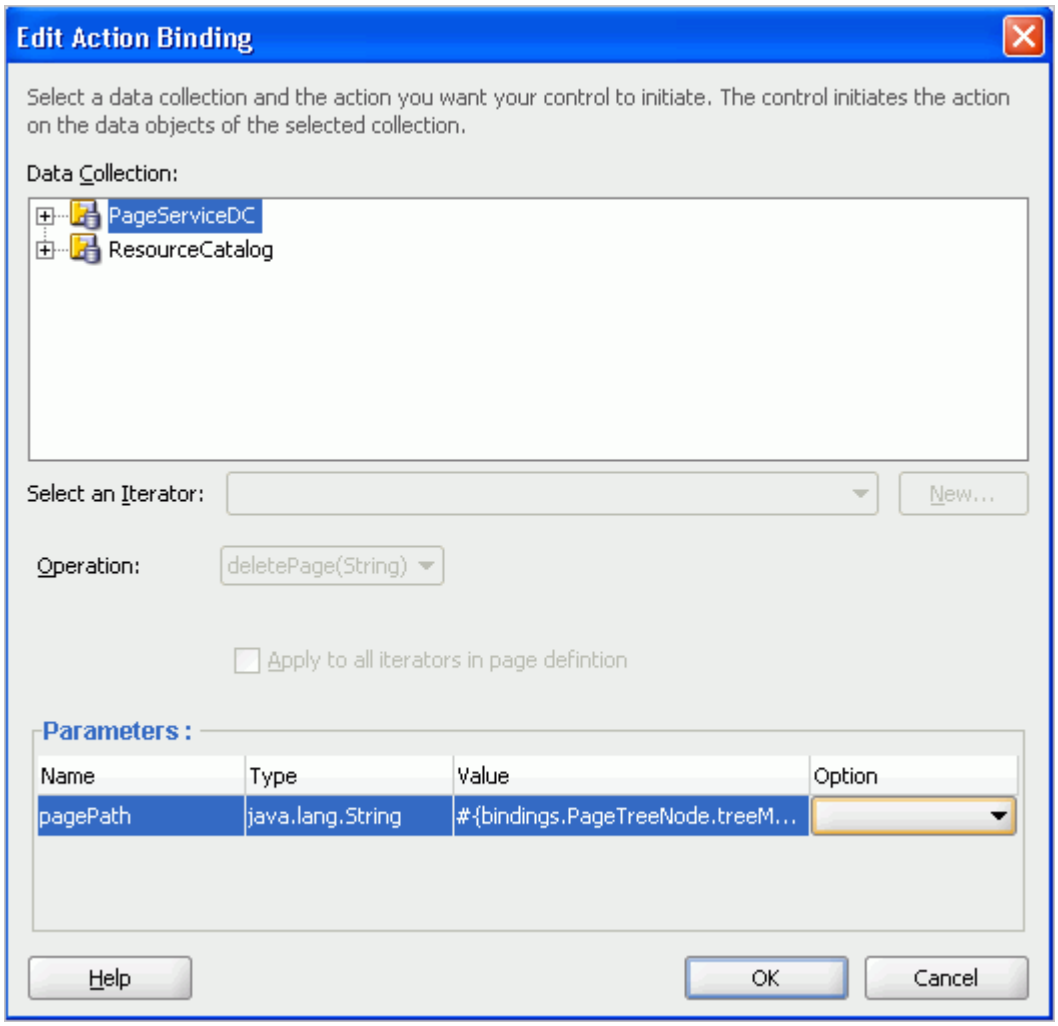
Figure 19–21 scopeName Parameter



6. Add a new column to the table you created at step 4.
7. To add a column for deleting pages at runtime, drag the **deletePage(string)** method and drop it in the new column.
8. From the **Create** menu, select **Methods**, and then select **ADF Link**.
9. In the Parameters section of the Edit Action Binding dialog, shown in Figure 19–22, specify the following value for the `pageName` parameter:

   ```
   #{bindings.PageTreeNode.treeModel.rowData.pagePath}
   ```

Figure 19–22 Edit Action Binding Dialog



10. Click **OK**.
11. From the Data Controls panel, expand **PageServiceDC**, **getPageTree()**, **PageTreeNode**, and then **Operations**.
12. Drag **Execute** and drop it on the page as an **ADF Button**.
13. In the Source view, set the button's `Text` attribute to `Refresh`.

    This provides a refresh feature on the list of pages to enable users to refresh the list after they have added a page or deleted one from the list.
14. Optionally, to display pages as links that invoke page Edit mode, view the page source and replace the `outputText` element in the table's page path column with a `goLink` as follows:

    Replace

    ```
    <af:outputText value="#{row.pagePath}"/>
    ```

    with

    ```
    <af:goLink destination="/faces#{row.pagePath}" text="#{row.title}"
    targetFrame="_top"/>
    ```

15. Save your page, and run it to your browser.
16. The created data control table should look something like Figure 19–23.

Figure 19–23 Sample Data Control Table

title	pageName	pagePath	creator	lastModified	hidden	
myPersonalPage	Page1.jspx	/oracle/webcenter/pa	steve	10/7/2008	false	deletePage
FinancePage	Page2.jspx	/oracle/webcenter/pa	steve	10/7/2008	false	deletePage
HRPage	Page3.jspx	/oracle/webcenter/pa	steve	10/7/2008	false	deletePage
MarketingPage	Page4.jspx	/oracle/webcenter/pa	steve	10/7/2008	false	deletePage
OperationsPage	Page5.jspx	/oracle/webcenter/pa	steve	10/7/2008	false	deletePage

19.8.2.2 How to View, Edit, and Delete Pages at Runtime

To view a page, follow the link generated from step 14 in the previous section. If the application has been secured properly, then an edit link should render on the page at runtime.

To delete a page, click the **deletePage** button in the row of the relevant page. This button was generated from in step 7 in the previous section. Click the Refresh button for the table to remove the deleted page row from the table.

For detailed information about how to view, edit, and delete pages at runtime, see the chapter "Creating, Editing, and Deleting Pages" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

19.9 Page Service Samples

The Custom Portal Demo sample illustrates some Page service functionality, including the following:

- How to add the Page - Create New task flow
- How to configure task flow parameters
- How to use an ADF template
- How to use custom styles
- How to manage pages

The Custom Portal Demo sample is posted on the Oracle WebCenter Suite 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at: `http://www.oracle.com/technetwork/middleware/webcenter/overview/index.html`

20

Enabling Runtime Editing of Pages Using Composer

Oracle WebCenter Portal's Composer provides components for controlling the application's design time at runtime behavior. This chapter describes how to add Composer components to your application page, thereby enabling runtime editing in the page.

For more information about Composer, see Chapter 18, "Introduction to Composer."

This chapter contains the following sections:

- Section 20.1, "Designing Editable Pages Using Composer Components"
- Section 20.2, "Designing Editable Pages Using Composer Components: Example"
- Section 20.3, "Populating Pages with Content"
- Section 20.4, "Troubleshooting Composer Problems"

20.1 Designing Editable Pages Using Composer Components

The Composer tag library provides design-time components that you can add to a page in Oracle JDeveloper to enable page editing at runtime. When you create a page with Composer components at design time, at runtime Composer provides options for entering page edit mode and modifying the page according to your requirements.

You can enable customizations in Framework applications and non-Framework applications. Within an application, you can enable customization of the following types of pages:

- A regular JSPX page, not based on a page template
- A JSPX template page
- A JSPX page based on a page template

For more information about Composer components and their attributes, see Section 18.6, "Composer Components."

This section explains how to add Composer components to a page at design time to make it editable at runtime. It contains the following subsections:

- Section 20.1.1, "How to Create a Customizable Page"
- Section 20.1.2, "How to Enable Runtime Customization Using a Page Customizable"
- Section 20.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button"

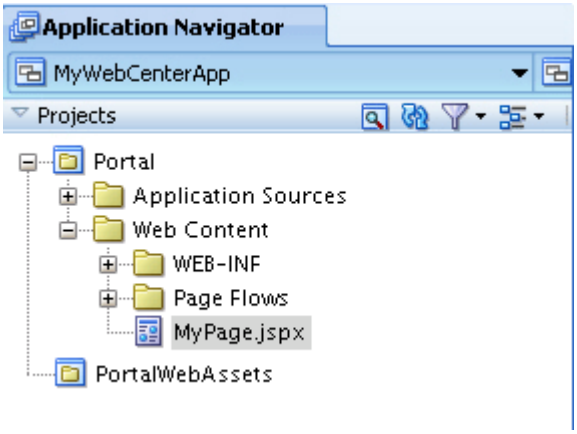
- Section 20.1.4, "How to Define Editable Areas of a Page Using Panel Customizable Components"
- Section 20.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable"
- Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components"
- Section 20.1.7, "How to Create a Page Template for Creating Customizable Pages"
- Section 20.1.8, "How to Enable Customization in a Populated Page"
- Section 20.1.9, "What Happens When You Add Composer Components"
- Section 20.1.10, "What Happens at Runtime"
- Section 20.1.11, "What You May Need to Know When Designing Editable Pages"

20.1.1 How to Create a Customizable Page

The steps for creating a customizable JSPX page in your Framework application are available in Section 5.3, "Adding Pages to a Portal."

When you create a new page in JDeveloper, it is listed in the Application Navigator under **Web Content** as shown in Figure 20–1. Additionally, the page is opened in the editor and becomes the active editor panel.

Figure 20–1 New Customizable Page (MyPage.jspx) Shown in the Application Navigator



Security Considerations

By default, a Framework application is configured with ADF security. A default user name and password (weblogic/weblogic1) are created automatically for you.

To enable users to edit a page in a secured application, you must grant Edit or Customize privileges on the page to the required users or roles. For more information about granting permissions, see Section 69.4, "Using the Role Manager Task Flow."

If ADF security is not enabled in your application, you can test how user privileges impact runtime customization capabilities by implementing security and configuring your application to authenticate users so that they have distinct identities. For the steps to implement a basic security model in your application, see Section 69.3, "Configuring ADF Security."

20.1.2 How to Enable Runtime Customization Using a Page Customizable

Adding a `Page Customizable` component to the page ensures that Composer is invoked when users switch to Edit mode of the page. When you add a `Page Customizable` component, some configuration files are updated automatically with the default Composer-specific settings. For more information, see Section 20.1.9, "What Happens When You Add Composer Components."

> **Note:** For considerations you must make before adding a `Page Customizable` to the page, see Section 20.1.11, "What You May Need to Know When Designing Editable Pages."

To add a Page Customizable component to the page:

1. Open a customizable JSPX page.
2. In the Component Palette, select **ADF Faces** and drag a **Panel Stretch Layout** component onto the page.

   > **Notes:** The `Panel Stretch Layout` component stretches the child in the `center` facet to fill all of the available space. This holds true when users resize the browser.

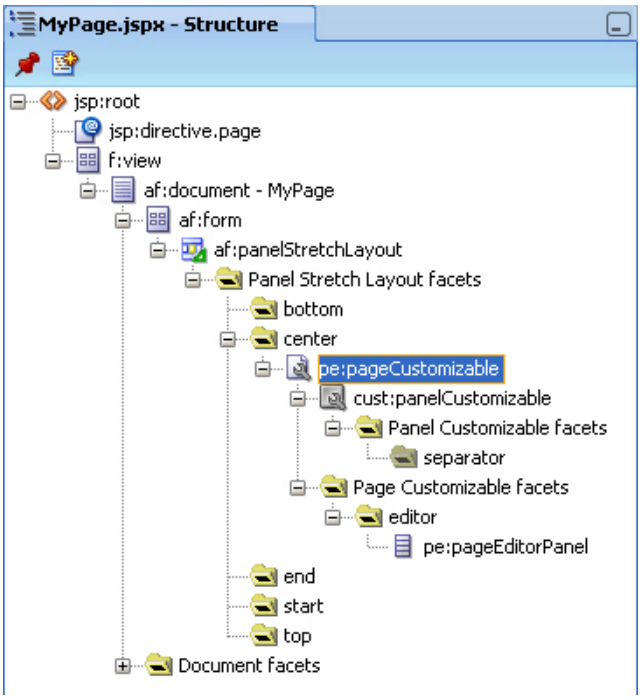
3. In the Component Palette, select **Composer**.
4. Add a **Page Customizable** component to the `center` facet of the `Panel Stretch Layout`.

   You must ensure that the `Page Customizable` is nested inside an `af:form` element. The `Page Customizable` is a rich client component that requires the rich client framework to function properly.
5. The required attributes for a `Page Customizable` component are populated with default values when you add the component to the page.

   You can define or modify attribute values by referring to Table B–1 in Section B.1, "Composer Component Properties."
6. Save the page.

   By default, a `Panel Customizable` component is added as a child component and the `Page Editor Panel` is added as a facet in the `Page Customizable` component, as shown in Figure 20–2.

Figure 20–2 Page Customizable Component



Example 20–1 shows the `pe:pageCustomizable` tag in the page source view.

Example 20–1 Page Customizable Component in the Page Source View

```
<pe:pageCustomizable id="pageCustomizable1"
  <cust:panelCustomizable id="panelcustomizable1"
                          layout="scroll"/>
  <f:facet name="editor">
    <pe:pageEditor id="pep1"/>
  </f:facet>
</pe:pageCustomizable>
```

20.1.3 How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button

To enable users to switch to Edit mode of a page easily, you must add a `Change Mode Link` or `Change Mode Button` component to the page.

Note: An alternative way to enable switching to Edit mode is by using the Change Mode API. For more information about this API, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

For things you should consider before adding a `Page Customizable` to the page, see Section 20.1.11, "What You May Need to Know When Designing Editable Pages."

To add a Change Mode Link or Change Mode Button component:

1. From the Component Palette, select **Composer**.

2. In the Structure window, within the `top` facet of the `Panel Stretch Layout` that you added in the previous section, drag a **Change Mode Link** or **Change Mode Button** component.

   You must ensure that the `Change Mode Link` or `Change Mode Button` is nested in an `af:form` element. The `Change Mode Link` or `Change Mode Button` component is a rich client component that requires the rich client framework to function properly.

   Notes:

   - If you have not used a `Panel Stretch Layout` on your page, then add the `Change Mode Link` or `Change Mode Button` component above the `Page Customizable` component in the Structure window. This ensures that the `Change Mode Link` or `Change Mode Button` is displayed properly at runtime.
   - Use a `Change Mode Link` or `Change Mode Button` only when you have a `Page Customizable` on the page. You may have problems running a page that contains a `Change Mode Link` or `Change Mode Button` but no `Page Customizable` component.

3. The required attributes for a `Change Mode Link` or `Change Mode Button` component are set by default when you add the component to the page.

   Optionally, you can set any other attributes by referring to Table B–2 in Section B.1, "Composer Component Properties."

   Note: In a secured application, it is recommended that you check all users' privileges and enable the Edit link or button on the page only for privileged users. Unauthenticated users who stumble into page Edit mode can change component properties, though the changes will not be saved.

   You can enable the Edit link or button for selected users by specifying an EL value for the `rendered` attribute on the `Change Mode Link` or `Change Mode Button` component. For more information, see Section B.1.2, "Change Mode Link and Change Mode Button."

   The `pe:changeModeLink` or `pe:changeModeButton` tag displays within the `af:form` tag in the Structure window, as shown in Figure 20–3.

Figure 20–3 Change Mode Link

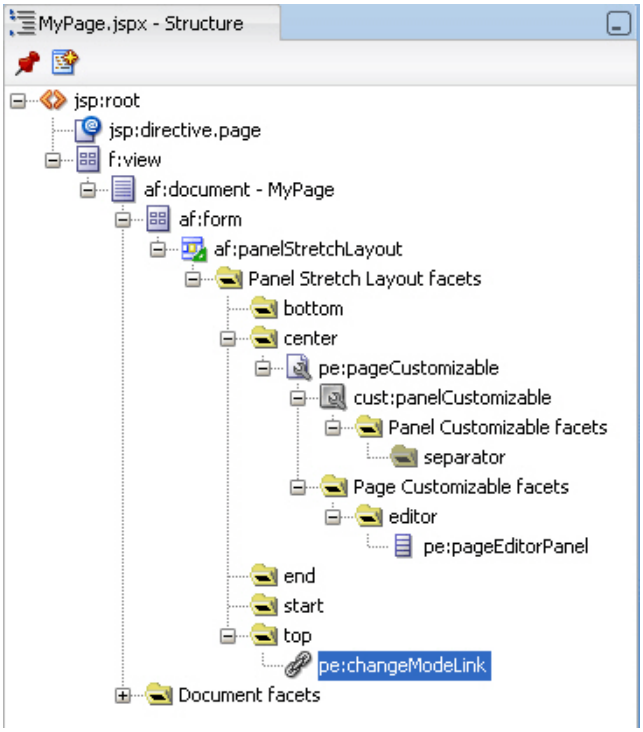
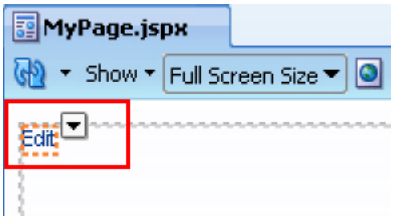


Figure 20–4 shows the `Change Mode Link` in the Design view of the page in JDeveloper.

Figure 20–4 Change Mode Link in Design View



20.1.4 How to Define Editable Areas of a Page Using Panel Customizable Components

The `Panel Customizable` component is required for page composition or content management tasks, such as adding, arranging, or removing portlets or regions. By default, one `Panel Customizable` component is automatically added as a direct child of the `Page Customizable` component. You can add more `Panel Customizable` components within this `Panel Customizable` component according to your requirements.

It is only within a `Panel Customizable` component that you can drag and drop components at runtime.

Note: For considerations you must make before adding a `Page Customizable` to the page, see Section 20.1.11, "What You May Need to Know When Designing Editable Pages."

To add a `Panel Customizable` component to the page:

1. From the Component Palette, select **Composer**.
2. Drag a **Panel Customizable** component to the Structure window and drop it at any suitable location within the form.

   You must ensure that the `Panel Customizable` is nested in an `af:form` element. The `Panel Customizable` component is a rich client component that requires the rich client framework to function properly.

   Notes:

   - A `Page Customizable` component contains one direct child `Panel Customizable` component by default. Do not add another `Panel Customizable` as a direct child component of the `Page Customizable`. If the `Page Customizable` has multiple child components, only the first child component is picked up while running the page.
   - Ensure that you do not delete the root `Panel Customizable` component on the page, because at runtime you can drop components *only* inside a `Panel Customizable` component.

3. The required attributes for a `Panel Customizable` component are set by default when you add the component to the page.

   Optionally, you can set any other attributes by referring to Table B–6 in Section B.1, "Composer Component Properties."

   Notes: If you select `stretch` layout for the `Panel Customizable`, then the first child component is stretched to fill up available space in the `Panel Customizable` component. Any other child components are ignored, though they are not removed from the page.

20.1.5 How to Enable Layout Customization for a Page Using a Layout Customizable

Use the `Layout Customizable` component to enable the runtime definition or modification of the layout of components on a page or an area of a page. Use this component only if you want to allow users to customize the layout at runtime. For static layouts, use an alternative component, such as a `Panel Group Layout` or `Panel Stretch Layout`.

To add a `Layout Customizable` component:

1. From the Component Palette, select **Composer**.
2. Drag a **Layout Customizable** component to the Structure window and drop it inside the `Panel Customizable` component.

   The target `Panel Customizable` component must be a child of the `Page Customizable` component.

   Ensure that the `Layout Customizable` is nested in an `af:form` element. The `Layout Customizable` is a rich client component and requires the rich client framework to function properly.

Note: You can delete the direct child `Panel Customizable` in the `Page Customizable` and add the `Layout Customizable` as a direct child of the `Page Customizable`. However, you must ensure that the `Page Customizable` has only one direct child component.

3. The required attributes for a `Layout Customizable` component are set by default when you add the component to the page.

   Optionally, you can set any other attributes by referring to Table B–3 in Section B.1, "Composer Component Properties."

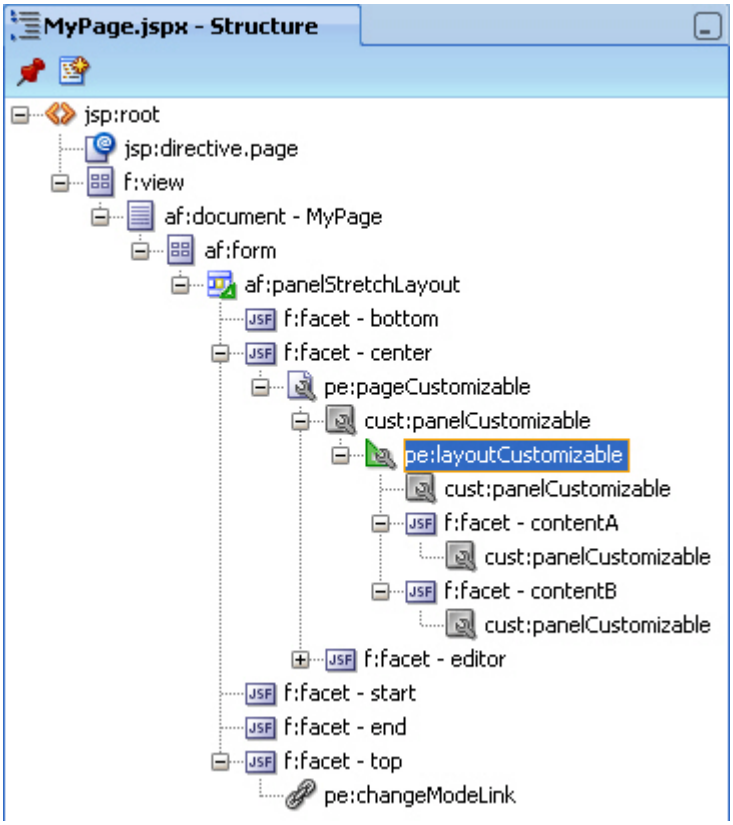
   Note: To ensure that the `Layout Customizable` component is clearly visible on the page at runtime, provide a descriptive label for the component by using the `Text` attribute.

   The `pe:layoutCustomizable` tag is located inside a `cust:panelCustomizable` tag in the Structure window as shown in Figure 20–5. A child `Panel Customizable` component is added by default in the `Layout Customizable` component. Additionally, a `Panel Customizable` component is added within each facet of the `Layout Customizable` component. These `Panel Customizable` components enable you to add content inside the `Layout Customizable` component at runtime.

   The `Panel Customizable` added as a direct child provides the main area—the central area in a layout at runtime. The `Panel Customizable` components added within the two default `Layout Customizable` facets provide the two content areas, `A` and `B`. When you select a predefined layout at runtime, these three areas are arranged to display content in the selected pattern. See Predefined Layout Types for more information about how the content is laid out for each layout type.

   Note: In a `oneColumn` layout, `A` and `B` are rendered only if the `Panel Customizable` components contain child components.

Figure 20–5 Layout Customizable Component



20.1.6 How to Enable Component Customization Using Show Detail Frame Components

When you want to enable customizations such as property editing, moving, minimizing, or removing components, you can drop a `Show Detail Frame` component inside a `Panel Customizable` component on the page. You can then add a component inside the `Show Detail Frame`.

> **Note:** Each `Show Detail Frame` component should have only one direct child component. If you add multiple child components, then only the first one is rendered. The other direct child components are not rendered at design time or runtime.
>
> If multiple components must be enclosed in a `Show Detail Frame`, then add a grouping component like `Panel Group Layout` or `Panel Customizable` to the `Show Detail Frame` component and then include the ADF Faces components or other content within this grouping component.

Use the `Show Detail Frame` component to enable customizations in View and Edit modes of the page. Changes made in View mode are available to that user only, and changes made in Edit mode are available to all application users.

To add a Show Detail Frame component to the page:

1. From the Component Palette, select **Composer**.

2. Drag a **Show Detail Frame** component to the Structure window and drop it inside a `Panel Customizable` component.

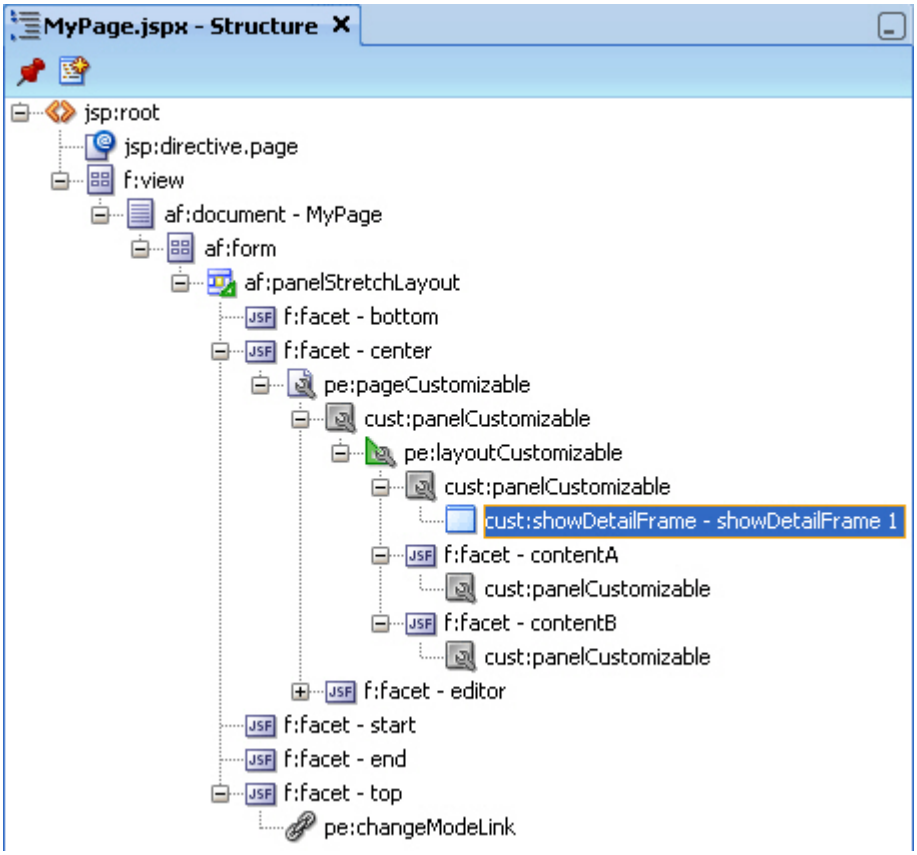
   The `Show Detail Frame` should be nested in a `Panel Customizable` component on the page.

3. The required attributes for a `Show Detail Frame` component are set by default when you add the component to the page.

   Optionally, you can set any other attributes by referring to Table B–7 in Section B.1, "Composer Component Properties."

   The `cust:showDetailFrame` tag is added inside the `cust:panelCustomizable` tag as show in Figure 20–6.

Figure 20–6 Show Detail Frame Component



For a better understanding of this type of task, see Section 20.2, "Designing Editable Pages Using Composer Components: Example."

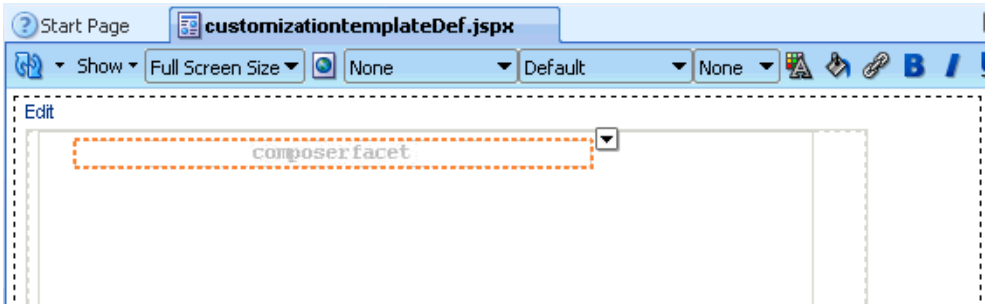
20.1.7 How to Create a Page Template for Creating Customizable Pages

If you plan to create many customizable pages in your application, you can base the pages on an ADF page template in which you have enabled customization. By adding Composer components to the template itself, you can save the effort required to add these components to each customizable page.

To create a page template and enable customization on it:

1. In the Application Navigator in JDeveloper, go to the Portal project of the application in which you want to create the template, right click the Portal project, and choose **New**.
2. In the New Gallery, expand **Web Tier,** select **JSF** and then **JSF Page Template,** and click **OK.**
3. In the Create JSF Page Template dialog, enter a file name for the template, select a layout, and declare a new facet.
4. Click **OK**.
5. In the template, add a **Change Mode Link** or **Change Mode Button** component above the `Panel Stretch Layout` component.
6. Add a **Page Customizable** component within the `center` facet of the `Panel Stretch Layout` component.
7. Delete the child `Panel Customizable` component from the `Page Customizable` and add a `Panel Group Layout` component in its place.
8. Set the `layout` attribute on the `Panel Group Layout` to `scroll`.
9. Add a `Facet Ref` component inside the `Panel Group Layout` and specify the same name that you used to declare the facet in step 3. The template displays the facet as shown in Figure 20–7.

Figure 20–7 New Page Template File



10. Save the template file.

You can now create customizable pages based on this template. Once you create a page, you can add a `Panel Customizable` or `Layout Customizable` component inside the facet displayed on the new page. You can then add page content within the `Panel Customizable` or `Layout Customizable` component.

20.1.8 How to Enable Customization in a Populated Page

When you have an existing ADF application with JSPX pages that are populated with content, and you want to enable customization, you can do so by moving all content inside a `Page Customizable` component.

You must first add the `Page Customizable`, then a `Layout Customizable`, and then the required hierarchy of `Panel Customizable` and `Show Detail Frame` components. Drag each of the existing components and drop them onto suitable locations inside the `Page Customizable`.

Notes:

- Ensure that the `Page Customizable` component has only one direct child component.

  When you add multiple direct child components, only the first child component is rendered at runtime. The first child component is stretched to fit the page. All other direct child components are ignored and not rendered on the page.

- Use a `Layout Customizable` component only if you want to allow users to customize the layout at runtime. For static layouts, use an alternative component, such as a `Panel Group Layout` or `Panel Stretch Layout`

- For best results, move components using the Structure window and *not* by editing the source code.

20.1.9 What Happens When You Add Composer Components

When you add a `Page Customizable` component to the page, the following configurations are performed automatically:

- A default Resource Catalog definition file, `default-catalog.xml`, is configured in the application. The `default-catalog.xml` file is located in the *`Application_Root`*`/Portal/src/portal` directory. To add components to the default Resource Catalog available to end users of your application, see Chapter 16, "Creating and Managing Resource Catalogs."

  In a Framework application, the `default-catalog.xml` file is located in the *`Application_Root`*`/Portal/public_html/oracle/webcenter/portalapp/catalogs` directory out-of-the-box, even without creating a page and adding a `Page Customizable` component.

- A Resource Catalog Viewer is configured for the application. At runtime, Composer provides users the option to add resources from this viewer to the page.

- The `web.xml` file available in the *`Application_Root`*`/`*`Project_Name`*`/public_html/WEB-INF` directory is updated to configure the MDS JSP provider.

- The `ComposerChangeManager` is configured in the application's `web.xml` file. For more information, see Section 22.11, "Configuring the Persistence Change Manager."

- When you create an application, a minimal `adf-config.xml` file is also created. When you add a `Page Customizable` to an application page, the required configuration is added to the `adf-config.xml` file. For example, change persistence is configured in the `adf-faces-config` section in this file. For more information, see Section 22.11, "Configuring the Persistence Change Manager."

- The `DataBindings.cpx` file in the *`Application_Root`*`/Portal/adfmsrc/portal` directory, is updated to enable the presence of task flows on the page.

- The page definition file is updated with the binding for the Composer task flow, which is available as part of the WebCenter Portal extension JAR file. Example 20–2 shows the code in the page definition file after a `Page Customizable` component is added to an application page.

Example 20–2 Page Definition File After Adding Page Customizable Component

```
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
                version="11.1.1.57.95" id="rcpagePageDef"
                Package="portal.pageDefs">
  <parameters/>
  <executables>
    <variableIterator id="variables"/>
    <taskFlow id="pageeditorpanel"
              taskFlowId="#{pageEditorBean.pageEditorPanel}"
              xmlns="http://xmlns.oracle.com/adf/controller/binding"/>
  </executables>
  <bindings/>
</pageDefinition>
```

See Also: Section B.2, "Composer-Specific Files and Configurations" for information about files that are created or modified when you add Composer components.

20.1.10 What Happens at Runtime

At runtime, users can perform all the tasks described Section 18.3, "Customizing Capabilities in Page View Mode" and Section 18.4, "Editing Capabilities in Design View in Page Edit Mode."

Each Composer component provides runtime capabilities that are described in Section 18.6, "Composer Components."

Note: Runtime customizations that you perform on a page are not carried over when you deploy the application to a target server.

20.1.11 What You May Need to Know When Designing Editable Pages

When adding Composer components to your customizable page, consider the following:

- To enable users to edit a page in a secured application, you must explicitly grant Edit or Customize privileges on the page to the required users or roles. For more information about granting permissions, see Section 69.4, "Using the Role Manager Task Flow."
- To enable runtime customization of components, add only one `Page Customizable` to a page.

  Note: Do *not* add a second `Page Customizable` component to your page. Only the first `Page Customizable` component is picked up when you run the page.

- Ensure that the `Page Customizable` component has only one direct child component.

  When you add multiple direct child components, only the first one is rendered at runtime. The first child component is stretched to fit the page. All other direct child components are ignored.

- Place all components you want to be customizable within the `Page Customizable` component.
- To enable runtime editing, you *must* ensure that the `ID` attribute is defined on all components on the page. Runtime editing of components that have no `ID` value is not supported in Composer.

  If your page includes components with no `ID` value, then you may encounter problems while editing the page in Composer.
- To enable View mode user customization, place a `Show Detail Frame` component within a `Panel Customizable` component.
- Use a `Show Detail Frame` to enclose a single child only. If you must enclose multiple components in a `Show Detail Frame`, then place a grouping component, such as a `Panel Group Layout` or `Panel Customizable`, within the `Show Detail Frame` component and then place the ADF Faces components or other content within this grouping component.
- Portlets need not be placed within `Show Detail Frame` components. Portlets come equipped with a header and display options that are similar to `Show Detail Frame` components.

20.2 Designing Editable Pages Using Composer Components: Example

In this example, assume you want to create a page that is customizable at runtime. The page is named `MyPage.jspx` in a Framework application named `MyWebCenterApp`.

To create a customizable page:

1. Create a Framework application named `MyWebCenterApp` by performing the steps in Section 5.1, "What Are Application Templates?."
2. Create a JSPX page named `MyPage.jspx` by performing the steps in Section 5.3, "Adding Pages to a Portal."
3. Add a `Panel Stretch Layout` to `MyPage.jspx`.

   Note: The `Panel Stretch Layout` component stretches the child in the center facet to fill all the available space in the browser. This is true even if you resize the browser. Therefore, by placing your components within this child component, you can ensure that the customizable portion of your page occupies the complete browser area.

4. Add a `Page Customizable` to the center facet by following the steps in Section 20.1.2, "How to Enable Runtime Customization Using a Page Customizable."
5. Set the border color of the `Page Customizable` to `blue` to differentiate the editable area from other noneditable areas.

   Under the Style category in the Property Inspector for the `Page Customizable`, click the Box tab and set the `Border Color` attribute to `Blue`.
6. Set the border color of child `Panel Customizable` to `red`.

   Under the Style category in the Property Inspector for the `Panel Customizable`, click the Box tab and set the `Border Color` attribute to `Red`.

7. Add a `Change Mode Link` to the top facet of the `Panel Stretch Layout` by performing the steps in Section 20.1.3, "How to Enable Switching Between Page Modes Using a Change Mode Link or Change Mode Button."

8. Inside the `Panel Customizable` that is a direct child of the `Page Customizable`, add a `Layout Customizable` component by following the steps in Section 20.1.5, "How to Enable Layout Customization for a Page Using a Layout Customizable."

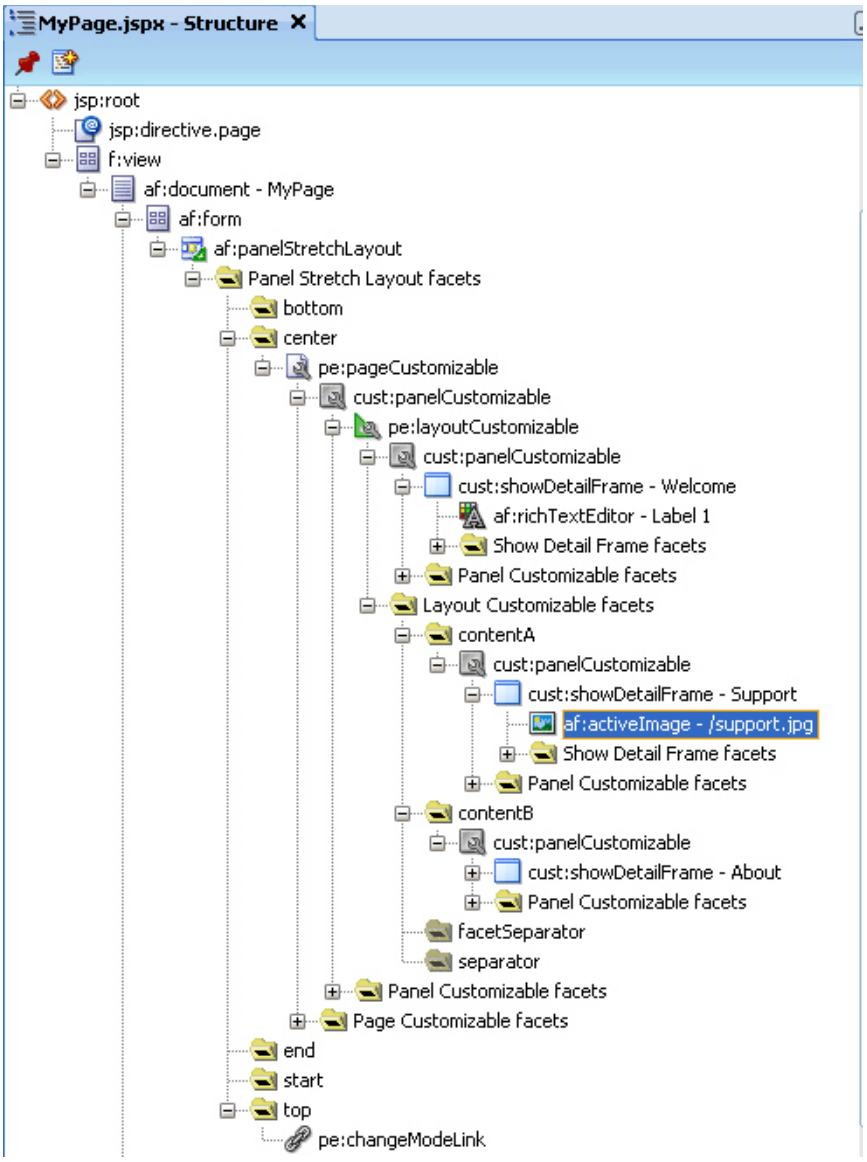
9. Add `Show Detail Frame` components inside each `Panel Customizable` on the page by performing the steps in Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components," and set the

10. For the sake of this example, add a `Rich Text Editor` and an `Image` component inside two of the `Show Detail Frame` components nested in the `Layout Customizable`.

    Drag and drop each of these components from the **ADF Faces** tag library to the required location on the page.

    The hierarchy of components on the page is as shown in Figure 20–8.

Figure 20–8 Component Hierarchy in MyPage.JSPX



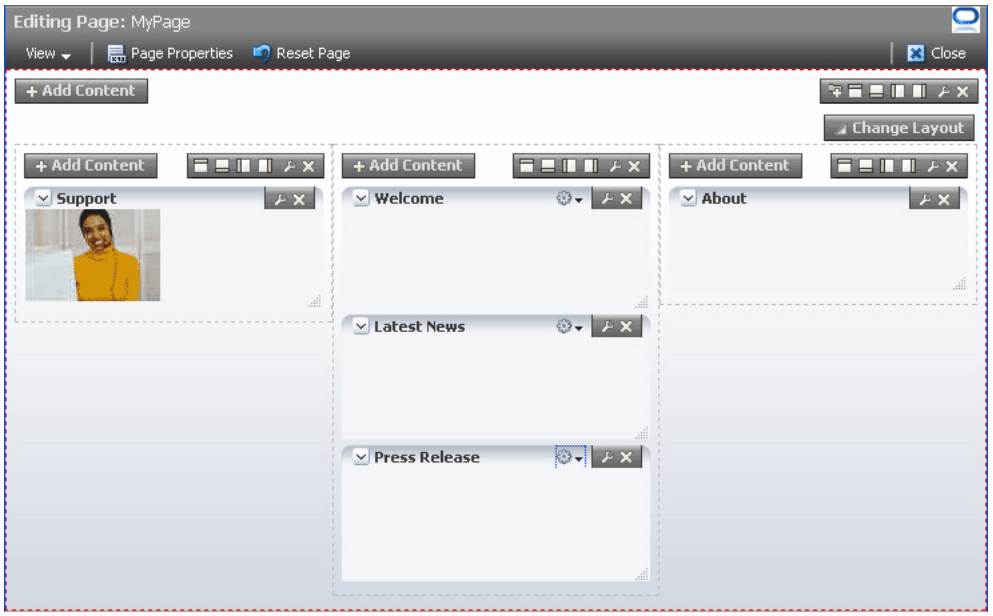
11. Grant Edit or Customize permission on the page to the required users and roles. For more information about granting permissions, see Section 69.4, "Using the Role Manager Task Flow."

12. Run `MyPage.jspx`.

    The page opens in View mode. Click the **Edit** link on the page to enter Edit mode. The page opens in Composer. In Composer, you can perform all editing tasks described in Section 18.4, "Editing Capabilities in Design View in Page Edit Mode."

    Figure 20–9 shows how the page appears in Edit mode at runtime.

Figure 20–9 MyPage Opened in Composer



Composer

****

If your application is not configured with ADF security, then to use this sample page in other examples in this guide, configure ADF security on the application.

To configure security in your application:

1. Configure ADF Security with form-based authentication, generating a default login page.

   For more information, see Section 69.3, "Configuring ADF Security."

2. Create three users `ahunold`, `sking`, and `jdoe`.

   For the detailed procedure, see Section 69.2, "Creating an Application Role."

When you run `MyPage.jspx`, a login screen is displayed. You can log in with any of the three user names that you created.

20.3 Populating Pages with Content

After you create an editable page with the required Composer components, you can populate the page with content just like a regular JSPX page. However, there are certain limitations and recommendations that you must be aware of when adding content to your Composer-enabled page.

Populating editable pages at design time is like populating any other ADF Faces page. You can drag and drop components from different areas of the IDE onto the page. You can add components like portlets, task flows, and ADF Faces components.

When you drag and drop a component anywhere inside a `Page Customizable` component, the `Id` attribute is set to a unique value. The `Id` attribute is required for editing a component and persisting its changed state. When you add a component to a page at runtime, the `Id` attribute is set automatically.

20.3.1 What You May Need to Know When Adding Content to the Page

Consider the following when adding content to your editable page:

- Add components inside `Panel Customizable` components that are nested in the `Page Customizable` component. This ensures that the components can be edited at runtime.
- You can include any Oracle ADF Faces component or task flow as child component of a `Show Detail Frame` component. However, portlets contain headers similar to those provided by `Show Detail Frame` components and can be added to `Panel Customizable` components directly. There are no additional benefits to including portlets in `Show Detail Frame` components.
- For components that you want to make selectable in Source view in Composer, ensure that the component is compliant with rich client framework and generates a client side component.
- If your page has the ADF Faces components `Output Text` and `Output Formatted` nested inside a `Page Customizable` component, then ensure that you set the `clientComponent` attribute value. If this attribute value is not set, then you may encounter errors while trying to move or rearrange components on the page at runtime.
- To consume portlets in your editable page, you must first register the portlet producers with your application. See Chapter 64, "Consuming Portlets" for details.

20.3.1.1 Considerations for Adding Task Flows

Users can personalize, customize, and edit task flows that are added to a customizable page. In a secured application, users with Customize permission on the task flow can also edit components on the task flow's page.

Consider the following points when adding task flows to your customizable page:

- To provide a consistent user experience in terms of look and feel, layout design, and interaction, and to avoid problems, such as too much white space, too many scroll bars, or slow, jerky, and unpredictable responses when displaying large data sets, ensure that the task flows are created by following the guidelines in Appendix D, "Guidelines for Creating Task Flows to Be Used in Composer-Enabled Pages."
- In a secured application, to enable users to view task flow content in Composer, ensure that the task flow has a `TaskFlowPermission` grant in the application's `jazn-data.xml` file, with at least `View` action provisioned. You must set this explicitly as it is not enabled by default.
- In a secured application, to enable users to edit components on the task flow's page, ensure that the task flow has a `TaskFlowPermission` grant in the application's `jazn-data.xml` file, with `Customize` action provisioned. You must set this explicitly as it is not enabled by default.
- Ensure that you specify valid values for all required parameters on the task flow.

For more information, see Section 0.1, "Implementing Task Flow Security."

20.4 Troubleshooting Composer Problems

This section provides information to assist you in troubleshooting problems you may encounter while using Composer.

Configuring ADF Logging for Composer

While creating your applications in JDeveloper, you can use the JDeveloper debugging tools to easily find errors in your web pages or page definition files. You can also set up the Java logger to display Java diagnostic messages. For detailed information about configuring logging, see the chapter titled "Testing and Debugging ADF Components" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*. To configure logging for Composer, perform the tasks in that guide and ensure that the `oracle.adf.view.page.editor` and `oracle.adfinternal.view.page.editor` packages are configured in the `logging.xml` file, with the desired logging level.

Problem

When you run a page, the following error displays:

```
java.lang.IllegalStateException: The expression
"#{bindings.pageeditorpanel.regionModel}"
(that was specified for the RegionModel "value" attribute of the region component
with id "pePanel")
evaluated to null. This is typically due to an error in the configuration of the
objects referenced by this expression. If it helps, the expression
"#{bindings.pageeditorpanel}" evaluates to "null". If it helps, the expression
"#{bindings}" evaluates to "view_untitled1PageDef". Now using an empty RegionModel
instead.
```

Solution

This error occurs if a page containing the `Page Customizable` component does not have the required task flow binding in its page definition file. Ensure that the page definition file contains the following valid entry under the `<executables>` node:

```
<taskFlow id="pageeditorpanel"
taskFlowId="#{pageEditorBean.pageEditorPanel}"xmlns="http://xmlns.oracle.com/adf/c
ontroller/binding"/>
```

This error may also occur if your page is based on a page template and that page template contains the `Page Customizable` component. In such a case, the `af:pageTemplate` tag does not contain the `value="#{bindings.pageTemplateBinding}"` attribute.

Ensure that the page definition file has the following entry under the `<executables>` node:

```
<page path="view.pageDefs.templateDef1PageDef"
id="pageTemplateBinding" Refresh="ifNeeded"/>
```

Problem

Users cannot switch to Edit mode. The Edit link (`Change Mode Link` or `Change Mode Button`) appears disabled.

Solution

The user may have only view privilege on the page. Ensure that the user has the edit privilege on the page. For the page template on which the page is based, it is sufficient to have only view privilege.

Problem

Users encounter an exception while wiring events in the Component Properties dialog in Composer.

Solution

This error occurs if your application page includes both ADF Data Visualization components and Composer components.

Register the `DvtElementObjectFactory` class with the Oracle ADF `FactoryManager` object by adding the following code to the application class that gets loaded the earliest:

```
...
import oracle.adfdt.model.dvt.objects.DvtElementObjectFactory;
import oracle.adfdt.model.managers.FactoryManager;
...
static
{
FactoryManager.getInstance().registerFactory(new DvtElementObjectFactory());
}
```

21

Extending Runtime Editing Capabilities Using Composer

This chapter describes how to use Composer's declarative and programmable extensibility mechanism to customize runtime capabilities to suit your business needs. It contains the following sections:

- Section 21.1, "Overview of Extensibility Options"
- Section 21.2, "Creating Composer Add-Ons"
- Section 21.3, "Creating Custom Property Panels"
- Section 21.4, "Extending the Expression Builder"
- Section 21.5, "Configuring Custom LOVs or Pickers for Task Flow Parameters"
- Section 21.6, "Configuring Event Handlers for Composer UI Events"
- Section 21.7, "Configuring Drop Handlers in the Resource Catalog"
- Section 21.8, "Defining Property Filters"
- Section 21.9, "Enabling Parameter Support on the Customization Manager Task Flow"
- Section 21.10, "Customizing the Composer Toolbar"
- Section 21.11, "Enabling Direct Select in Design View"
- Section 21.12, "Troubleshooting Problems with Composer Extensibility Features"

21.1 Overview of Extensibility Options

Composer provides a framework on which to build customizable application pages. In addition to its default capabilities, you can extend the Composer framework to augment the runtime capabilities available to end users. You must configure the extensions in your application's `adf-config.xml` file and the Composer extension file, `pe_ext.xml`. For information about these files, see Section 21.1.8, "Configuration Files."

This section describes the options available for extending Composer runtime capabilities declaratively. It contains the following subsections:

- Section 21.1.1, "Composer Add-Ons"
- Section 21.1.2, "Composer Custom Property Panels"
- Section 21.1.3, "Composer Events Handlers"
- Section 21.1.4, "Drop Handlers."

- Section 21.1.5, "Component Property Filters"
- Section 21.1.6, "Customization Manager"
- Section 21.1.7, "Composer Toolbar Customization"

21.1.1 Composer Add-Ons

The default view of the Composer toolbar (Figure 21–1) includes the following buttons:

- **View**: Allows the user to switch between Design and Source view.
- **Page Properties**: Opens the Page Properties dialog, allowing the user to edit metadata for the host page.
- **Reset Page**: Reverts any changes made to the page in the current edit session.
- **Close**: Exits Design view and returns the user to the standard UI.

Figure 21–1 Default Add-Ons on the Composer Toolbar



Composer

****

The Page Properties and Reset Page buttons are examples of add-ons. Click these buttons to display panels for editing page properties and resetting page customizations. Typically, add-ons are custom task flows that are rendered as buttons on the Composer toolbar in page Edit mode. You can create add-ons that appear along with the Page Properties and Reset Page add-ons. For example, you can create an add-on to display page revision history so that it displays a Revisions button on the Composer toolbar. Clicking this button would display the page's revision history. You can also replace the Page Properties and Reset Page add-ons with custom add-ons that you create.

The process of configuring custom add-ons includes creating the task flows, packaging them into JAR files, and defining them in the Composer extension file. For more information, see Section 21.2, "Creating Composer Add-Ons."

21.1.2 Composer Custom Property Panels

The Component Properties dialog displays categories of attributes on different tabs. Each tab can be referred to as a property panel. The default Component Properties dialog in Composer is analogous to the Oracle JDeveloper Property Inspector. You can create and register custom property panels for a component, populate them with component properties, and display them as tabs along with the default tabs in the Component Properties dialog.

The process of configuring custom property panels includes creating them as task flows, packaging them into JAR files, and defining them in the Composer extension file. For more information, see Section 21.3, "Creating Custom Property Panels."

21.1.3 Composer Events Handlers

Composer provides an intuitive user interface for editing pages at runtime. This includes such UI components as the Save, Close, and Delete buttons. When a user

clicks a button or icon in Composer, an event handler ensures that a specific action is performed. An event handler is the code that is called back by Composer when a Composer event is invoked. Each UI event in Composer is associated with an event handler. Sometimes it may be necessary to augment Composer's innate capabilities by performing a different action or multiple actions on invoking an event. For example, when a user clicks Save, in addition to the save operation that Composer provides by default, you might want to configure the application to perform additional tasks such as cleaning up cached information and connections to resources. You can accomplish this with event handlers. For more information, see Section 21.6, "Configuring Event Handlers for Composer UI Events."

21.1.4 Drop Handlers

The Resource Catalog provides resources that users can add to their pages. An Add link next to a resource name enables users to add it to the page. Composer provides *drop handler*s to handle the add operation in the catalog. By default, a drop handler is configured for each resource in the catalog. If you want to provide complete control of the drop action to the resource, you can create additional drop handlers for that resource. The Add link then displays a context menu with different options for adding the resource to the page. You can create one or more drop handlers to handle different flavors for resources in your catalog.

You can add the following types of drop handlers:

- Java classes registered with Composer and called when users click an Add link in the Resource Catalog. In this case, the resource is added to the page based on the data flavor selected.
- Task flows registered with Composer and called when users click an Add link in the Resource Catalog. In this case, a task flow is invoked, which enables users to decide what information from the resource must be displayed on the page.

For more information, see Section 21.7, "Configuring Drop Handlers in the Resource Catalog."

21.1.5 Component Property Filters

The Component Properties dialog displays properties of a selected component. By default, Composer filters certain component properties and displays a subset of properties to users. You can define filters declaratively to further hide properties that the user need not see, or to show properties that are hidden. For more information, see Section 21.8, "Defining Property Filters."

21.1.6 Customization Manager

Customization Manager is a task flow that enables users to download, upload, reset, and delete application customizations on objects like pages and task flows. You can configure the Customization Manager either in Composer or outside of it, on some administrative page. For information about configuring and using Customization Manager, see Section 21.9, "Enabling Parameter Support on the Customization Manager Task Flow."

21.1.7 Composer Toolbar Customization

The default Composer toolbar displays elements on two rows. The first row displays the page name and status indicator, and the second row displays the View menu, Page Properties button, Reset Page button, and Close button, as shown in Figure 21–2.

Figure 21–2 Composer Toolbar with Default Elements



You can customize the toolbar by adding, deleting, or rearranging elements. You can also override existing elements with custom elements. For example, you can remove the message showing the page name if you do not want users to see the name of the page they are editing.

For information about customizing the Composer toolbar, see Section 21.10, "Customizing the Composer Toolbar."

21.1.8 Configuration Files

Before you start with the extensibility configurations described in this chapter, there are two important configuration files that you must know about. Most of the extensions discussed in this chapter are defined in these files:

- Composer extension file (`pe_ext.xml`)

  The Composer extension file, `pe_ext.xml`, enables you to extend the editing capabilities provided by Composer. Within this file you can add elements to register new Composer add-ons and custom property panels, selectively render panels, register event handlers, and define property filters. The `pe_ext.xml` file is not available in your application by default. You must create it the first time you perform such tasks as including add-ons, property panels, or event handlers. Create this file in the `META-INF` directory under the project's Web context root or in the *`application_home/project`*`/src/META-INF` directory. When you run the application, the `pe_ext.xml` file is picked up from the JAR file included in the application classpath. Your application can include more than one extension file. However, you must ensure that the JARs containing the extension files are available on the application classpath so that the `pe_ext.xml` files are picked up for processing. Every JAR with a `pe_ext.xml` in its `META-INF` folder is processed, and the Composer extensions are loaded and combined. For information about the different elements you can define in `pe_ext.xml` to extend Composer capabilities, see Section B.2.1, "pe_ext.xml."

- Application's `adf-config.xml` file

  The `adf-config.xml` file specifies application-level settings that are usually determined at deployment and often changed at runtime. When you perform such tasks as registering new add-ons and custom property panels in Composer, or creating customization layers, you must add appropriate entries in the `adf-config.xml` file. The `adf-config.xml` file is created automatically when you create an application, and when you add a `Page Customizable` component to the page, certain configurations are added to this file.

  For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

21.2 Creating Composer Add-Ons

Composer provides the following default add-ons for runtime editing:

- Page Properties

This dialog opens when users click the Page Properties button on the Composer toolbar. The Page Properties dialog displays the current page's properties and enables users to modify property values.

- Reset Page

  The Reset Page dialog opens when users click the Reset Page button on the Composer toolbar. The Reset Page dialog enables users to remove application customizations made to a page and reset it to a previously-saved version or to its original out-of-the-box state. For more information, see Section 18.4.11, "Reset Page."

In addition to these, you can register new add-ons with Composer. For example, you can create an add-on to display page revision history so that it displays a Revisions button on the Composer toolbar. Clicking this button would display the page's revision history.

This section contains the following topics:

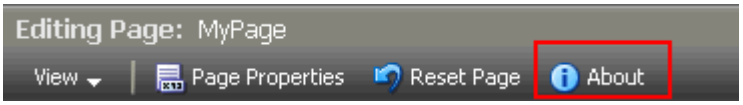
- Section 21.2.1, "How to Create and Register Add-Ons"
- Section 21.2.2, "What Happens at Runtime"
- Section 21.2.3, "How to Exclude Composer Default Add-Ons"
- Section 21.2.4, "How to Display the Customization Manager Add-On"
- Section 21.2.5, "How to Selectively Display Add-Ons"

21.2.1 How to Create and Register Add-Ons

You can create and register custom task flows that can be invoked from buttons on the Composer toolbar. All registered add-ons have an associated button that displays on the toolbar.

This section steps through the procedure of creating an add-on and registering it with Composer. It includes an example that demonstrates how to create an add-on that displays information about the application. The example add-on renders an **About** button on the Composer toolbar (Figure 21–3) that users can click to invoke a task flow that contains information about the application.

Figure 21–3 About Button on the Composer Toolbar



This section contains the following subsections:

- Section 21.2.1.1, "Creating an Add-On Task Flow"
- Section 21.2.1.2, "Registering Add-Ons with Composer"
- Section 21.2.1.3, "Registering Add-Ons in adf-config.xml"

21.2.1.1 Creating an Add-On Task Flow

Composer add-ons are task flows you create using JSPX pages or page fragments.

To create an add-on:

1. In your application project, create a JSFF file called `custompanelview.jsff`:

a. From the **File** menu, select **New**.

b. In the New Gallery dialog, expand **Web Tier**, select **JSF**, then **JSF Page** or **JSF Fragment**.

c. Click **OK**.

2. Design the fragment by adding code similar to what is shown in Example 21–1.

Example 21–1 Sample code in the JSFF Fragment

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <af:panelGroupLayout id="pnlgrp1" layout="vertical" halign="center">
    <af:spacer id="sp1" height="20" />
    <af:image id="customimage"
              source="/images/DemoLogo.jpg"
              shortDesc="Demo Logo"/>
    <af:spacer id="sp2" height="20"/>
    <af:outputText id="output1" value="This is a sample shopping cart application
based on Oracle ADF Framework.
It also uses the Oracle WebCenter Portal: Framework to enable collaboration,
application customization and user customization features."/>
    <af:spacer id="sp3" height="20"/>
    <af:outputText id="output2" value="Build : 11.1.1"
                   inlineStyle="font-weight:bold;"/>
  </af:panelGroupLayout>
</jsp:root>
```

Note: At runtime, the add-on panel is automatically sized to fit the content in this fragment.

3. Create a task flow definition called `custom-panel-task-flow`:

a. From the **File** menu, choose **New**.

b. In the New Gallery dialog, expand **Web Tier**, select **JSF**, then **ADF Task Flow**.

c. Click **OK**.

4. Drop the `custompanelview.jsff` fragment that you created onto the task flow definition.

See Also: "Getting Started with ADF Task Flows" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for information about creating task flows.

5. Save the task flow definition file.

Optionally, if you create the task flow in one application but want to consume it in another application, you must first package the task flow in an ADF library and add the resulting JAR in the consuming application.

To package the task flow in an ADF library:

1. Create a deployment profile for the task flow:

a. Right-click **Portal** project and choose **New**.

b. In the New Gallery, expand **General**, select **Deployment Profile**, and then **ADF Library JAR File**, and click **OK**.

c. In the Create Deployment Profile -- ADF Library JAR File dialog, enter a name for your deployment profile and click **OK**.

d. In the ADF Library JAR Deployment Profile Properties dialog, click **OK**.

e. In the Project Properties dialog, click **OK**.

2. In the Application Navigator, right-click the project folder, choose **Deploy**, *deployment profile name*, **to**, and then choose **to ADF Library JAR file**. This creates a **deploy** folder including the JAR file, in the project folder located at `<Application_Root>\Portal\deploy\`.

You can add this JAR file to any application in which you want to consume the add-on.

21.2.1.2 Registering Add-Ons with Composer

After you create the task flows, you must register them with Composer so that they are displayed on the Composer toolbar along with the default options.

To register an add-on with Composer:

1. If it does not already exist, create the Composer extension file, `pe_ext.xml` in the `META-INF` directory under the project's Web context root (for example, in the *APPLICATION_HOME*`\Portal\src\META-INF` directory):

   a. From the **File** menu, select **New**.

   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.

   c. Click **OK**.

   Name the file `pe_ext.xml`.

2. Add an `<addon-config>` element in the file, with a nested `<panels>` element.

3. Add one `<panel>` element for each task flow that you want to register as an add-on.

   Any number of panels can be declared under the `<panels>` element in the extension file.

   Example 21–2 shows the code of the extension file with a `<panel>` entry.

Example 21–2 Composer Extension File

```
<?xml version="1.0" encoding="US-ASCII" ?>
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <panels>
      <panel name="oracle.custom.panel" title="About"

icon="http://myforums.oracle.com/jive3/images/question-pts-available-16x16.gif"

taskflow-id="/WEB-INF/custom-panel-task-flow.xml#custom-panel-task-flow" />
    </panels>
  </addon-config>
</pe-extension>
```

For more information about the `addon-config` and other nested elements, see Section B.2.1.1, "addon-config."

4. Optionally, register event handlers for the custom panel by adding an `event-handlers` element within the `panel` element as shown in the following example:

```
<event-handlers>
  <event-handler event="close">
    oracle.custom.TaskFlowEventHandler
  </event-handler>
</event-handlers>
```

   For more information, see Section 21.6, "Configuring Event Handlers for Composer UI Events."

5. Save the `pe_ext.xml` file.

21.2.1.3 Registering Add-Ons in adf-config.xml

To register an add-on, you must add a reference to it in the application's `adf-config.xml` file. Add the `addon-panels` entry to define new add-ons.

To register add-ons in the `adf-config.xml` file:

1. Open the application's `adf-config.xml` file, located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the following namespace within the `adf-config` element in the file:

```
xmlns:pe="http://xmlns.oracle.com/adf/pageeditor/config"
```

3. Add a `<pe:page-editor-config>` entry with the namespace and include `<pe:addon-panels>` inside it.

   When you register a custom add-on, the default add-ons are not displayed in Composer by default. To display all the default add-ons, you must set the `show-default-addons` attribute on the `<pe:addon-panels>` tag to `true`. The default value for this attribute is `false`.

   Within `<pe:addon-panels>`, add `<pe:addon-panel>` entries for the new panels, as shown in Example 21–3.

   The `name` attribute must contain the name you used to register the panel in the Composer extension file.

Example 21–3 New Add-On Referenced in adf-config.xml

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:addon-panels show-default-addons="true">

    <pe:addon-panel name="oracle.custom.panel" />

  </pe:addon-panels>
</pe:page-editor-config>
```

> **Note:** If you do not specify any `<addon-panel>` entries under `<addon-panels>`, then only the default options are displayed in Composer.

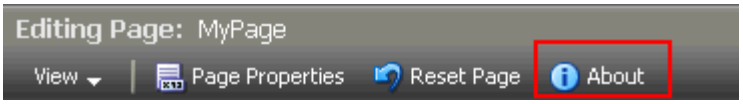
For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

21.2.2 What Happens at Runtime

Custom add-ons that you register with Composer are rendered on the Composer toolbar along with the default add-ons.

In the example, an About button is rendered on the Composer toolbar (Figure 21–4).

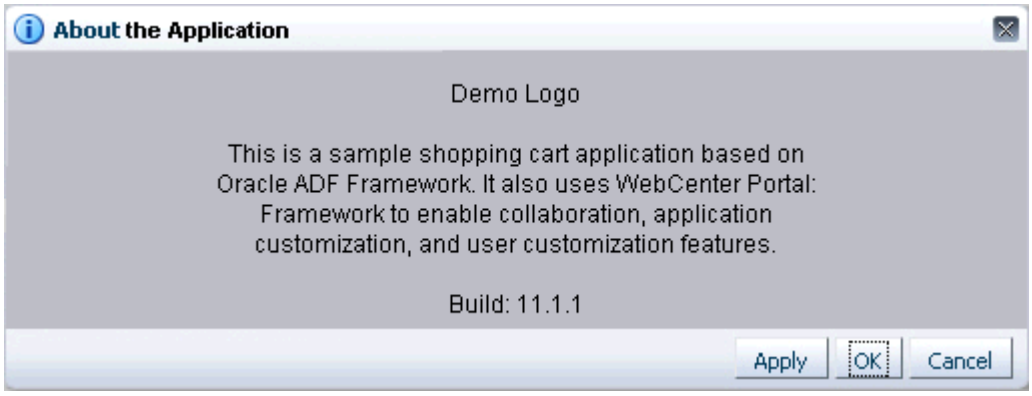
Figure 21–4 About Button in Composer Toolbar



Clicking this button displays the About the Application task flow (Figure 21–5).

> **Note:** The add-on panel is automatically sized to fit the content inside the task flow.

Figure 21–5 About the Application Task Flow



21.2.3 How to Exclude Composer Default Add-Ons

You can choose to show or hide the Page Properties, Reset Page, and Customization Manager buttons on Composer toolbar. To hide any of these add-ons, you must set the `show-default-addons` attribute on `<pe:addon-panels>` to `true`, add an entry for the default add-on you want to hide, and set the `rendered` attribute on that add-on to `false`.

Example 21–4 shows the code to hide the Page Properties add-on.

Example 21–4 The adf-config.xml File with the Reset Page Option Excluded

```
<pe:addon-panels show-default-addons="true">
  <!-- Hide the Reset Page add-on -->
  <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset"
rendered="false"/>

  . . .

</pe:addon-panels>
```

The excluded add-on (Reset Page) is not displayed on the Composer toolbar. The other default add-ons are displayed on the toolbar in the default order.

21.2.4 How to Display the Customization Manager Add-On

The Customization Manager is a Composer add-on that enables users to manage application customizations on task flows, pages, and page fragments on a given page. For more information, see Section 18.4.14, "Manage Application Customizations."

The Customization Manager add-on is available in the Composer library, but is not rendered by default. You can enable it by configuring it in your application's `adf-config.xml` file using a `<pe:addon-panel>` element inside `<pe:page-editor-config>`, as shown in the following example:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:addon-panels show-default-addons="true">
    <pe:addon-panel
name="oracle.adf.pageeditor.addonpanels.customization-manager"/>

    . . .

  </pe:addon-panels>
  . . .
</pe:page-editor-config>
```

This configuration ensures that the Composer toolbar displays the Customization Manager button as shown in Figure 21–6.

Figure 21–6 Customization Manager Button



21.2.5 How to Selectively Display Add-Ons

Depending on your business requirement, you may need to selectively hide the add-ons available to different users in Composer. To selectively hide an add-on, you must set the `show-default-addons` attribute on `<pe:addon-panels>` to `true`, add an entry for the default add-on you want to hide, and set the `rendered` attribute for that add-on to `false` using an EL expression, as shown in Example 21–5.

Example 21–5 rendered Attribute Setting in the adf-config.xml File

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:addon-panels show-default-addons="true">
    <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset"
                    rendered="#{securityBean.userInPageResetRole}" />
    . . .
  </pe:addon-panels>
  . . .
</pe:page-editor-config>
```

In this example, the `securityBean` backing bean returns either `true` or `false`, depending on the role of the logged-in user. If the returned value is `false`, Composer does not display the Reset Page button on the Composer toolbar. If the returned value is `true`, Composer displays the Reset Page button.

Note: If you hide the Reset Page button on the Composer toolbar, the Reset Task Flow button displayed while editing task flow content is also hidden.

The default value for `rendered` is `true`. That is, if you do not specify a `rendered` attribute for an add-on, the add-on is always displayed.

21.3 Creating Custom Property Panels

Composer displays the properties of a component in the Component Properties dialog when a user clicks the **Edit** icon on the component. The Component Properties dialog provides a series of tabs. Each tab displays a group of related attributes. The attributes have associated values that control a component's behavior and visual style properties. For example, the Style tab displays the component's style-related properties, such as width, height, and background color.

Similarly, when a user clicks the **Page Properties** button, a Page Properties dialog opens with its own series of tabs. These tabs contain display-related page properties, page parameters, and security settings.

You can create and register custom property panels to render along with the tabs displayed in the Component Properties or Page Properties dialog. In addition, you can remove the default panels or replace them with custom property panels. For example, you can develop a friendlier property panel for an `Image` component by displaying a picker for its `Source` property. This would make it easier for users to select an image from the available options.

This section describes how to create custom property panels. It also describes how to exclude, override, and selectively render default property panels. It contains the following subsections:

- Section 21.3.1, "How to Create and Register Custom Property Panels"
- Section 21.3.2, "What Happens at Runtime"
- Section 21.3.3, "How to Override Default Property Panels"
- Section 21.3.4, "How to Exclude Default Property Panels"
- Section 21.3.5, "How to Selectively Render Property Panels"
- Section 21.3.6, "How to Display Properties and Parameters in a Custom Property Panel"

21.3.1 How to Create and Register Custom Property Panels

Creating a custom property panel is similar to creating an add-on. That is, you create custom property panels as task flows and register them in the Composer extension file. However, while Composer add-ons are configured using the `addon-panels` element in the `adf-config.xml` file, custom property panels are configured using the `property-panels` element in the extension file itself.

You can configure a custom property panel to display in the Component Properties dialog always. Alternatively, you can configure the panel to display only when a particular component or task flow is selected for editing.

This section describes how to create and register a custom property panel. It contains the following subsections:

- Section 21.3.1.1, "Creating a Custom Property Panel"

- Section 21.3.1.2, "Registering a Custom Property Panel for a Component"
- Section 21.3.1.3, "Registering a Custom Property Panel for a Task Flow"

21.3.1.1 Creating a Custom Property Panel

Property panels provide a means of editing page or component properties. For example, a user can click the **Edit** icon on a selected task flow and modify its parameter values and change its visual attributes in the Component Properties dialog.

Composer enables you to associate property panels with components and task flows. When a user clicks the **Edit** icon on the component or task flow, Composer opens the Component Properties dialog and displays the custom property panels you associated with the object along with the default property panels.

The steps for creating a custom property panel and declaring it in the Composer extension file are similar to those for creating and declaring Composer add-ons. For detailed information, see Section 21.2.1.1, "Creating an Add-On Task Flow" and Section 21.2.1.2, "Registering Add-Ons with Composer."

Example 21–6 shows a sample property panel declaration in the `pe_ext.xml` file.

Example 21–6 Custom Property Panel Declaration in pe_ext.xml

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <panels>
      <panel name="sample-panel" title="Sample Panel"
             taskflow-id="/WEB-INF/sample_task_flow.xml#sample_task_flow"/>
    </panels>
  </addon-config>
</pe-extension>
```

After you create a custom panel and declare it in the `pe_ext.xml` file, you must configure it using the `<property-panels>` element, as shown in Example 21–7. This ensures that the panel displays automatically in the Composer Component Properties and Page Properties dialog.

Example 21–7 Custom Property Panel Configuration in pe_ext.xml

```
<addon-config>
  <panels>
    . . .
  </panels>
  <property-panels>
    <property-panel name="samplepanel">
      <panel name="sample-panel" rendered="true"/>
    </property-panel>
  </property-panels>
</addon-config>
```

Use one `<panel>` element to register each custom panel. A `<panel>` corresponds to a tab in the Component Properties dialog. You can add any number of `<panel>` elements.

21.3.1.2 Registering a Custom Property Panel for a Component

When you register a custom panel in the extension file, you can associate the panel with a component or a task flow. The custom property panel then renders only for that component or task flow. This section describes how to register a property panel for a specific component.

To register a property panel for a component:

1. Create a Composer extension file, `pe_ext.xml`, if it does not already exist.

   For information about creating the extension file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

2. Add a `<property-panels>` element inside the `<addon-config>` section in the `pe_ext.xml` file.

3. Add a `<property-panel>` declaration within the `<property-panels>` element.

   You can have multiple `<property-panel>` entries.

4. Within the `<property-panel>` element, add a `<component>` element to specify the runtime class name of the component (optional) and a `<panel>` element to specify the name you used to declare the panel in the `<addon-config>` section of the file.

   Example 21–8 shows a custom property panel that is associated with a `Command Button` component by specifying the component's fully qualified class name. For information about Oracle ADF components and their runtime classes, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Example 21–8 Code to Register a Property Panel for a Component

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <property-panels>
      <property-panel name="cmdbtn">

<component>oracle.adf.view.rich.component.rich.nav.RichCommandButton</component>
        <panel name="prop.panel.cmdbtn" />
        <panel name="prop.panel.generic" />
      </property-panel>
    </property-panels>
  </addon-config>
  . . .
</pe-extension>
```

> **Note:** When registering a property panel, if you do not associate it with a component or task flow (discussed in the next section), the registered panel is rendered at all times in the Component Properties and Page Properties dialogs.
>
> To configure multiple property panels for a component, you can include multiple `<panel>` elements within a `<property-panel>` element. For more information about the `<property-panels>` element and its nested elements, see Section B.2.1.2, "property-panels."

All the default panels, such as Child Components, Style, and Parameters, are displayed along with the custom panels you define.

21.3.1.3 Registering a Custom Property Panel for a Task Flow

You can define a custom property panel for a task flow by registering it with the Composer extension file. Example 21–9 shows the sample code used to register a custom property panel for a task flow.

A custom property panel registered for a specific task flow appears only when its associated task flow is selected. Otherwise, default property panels appear.

> **Note:** Use task flow-specific custom property panels only to customize task flow parameters or any other aspect related to how the task flow works.
>
> A custom property panel does not function if it is associated with a task flow rendered using the Oracle JSF Portlet Bridge.

To register a property panel for a task flow:

1. Add a `<property-panels>` element within the `<addon-config>` element in the `pe_ext.xml` file.

   For information about creating the extension file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

2. Add a `property-panel` declaration within this.

   You can have multiple `property-panel` entries.

3. Add `taskflow-id` and `panel` elements within the `property-panel` element.

   Add the `taskflow-id` element to specify the task flow name. Add the `panel` element to specify the name you used to declare the property panel in the `addon-config` section of the file.

   Example 21–9 shows a custom property panel associated with a `dashboard` task flow.

Example 21–9 Code to Register a Property Panel for a Task Flow Instance

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <property-panels>
      <property-panel name="dashboard">
        <taskflow-id>/WEB-INF/dashboard-taskflow#prop-panel</taskflow-id>
        <panel name="dashboard.prop-panel" />
      </property-panel>
    </property-panels>
  </addon-config>
  . . .
</pe-extension>
```

Notes:

- If you do not associate a custom property panel with a component or task flow (using the `<component>` or `<taskflow-id>` tag), then the registered panel is rendered for all pages, task flows, and components in the Component Properties and Page Properties dialogs.

  You can register multiple `panel` elements within a `property-panel` element. For more information about the `property-panels` element and its nested elements, see Section B.2.1.2, "property-panels."

- To enable end users to edit the attributes and parameters on a custom property panel, in the panel's task flow fragment you must add a `Show Property` components for each property that you want to expose. The value attribute on a `Show Property` component is tied to a managed bean that contains the logic to read and write values for the properties and parameters. For more information, see Section 21.3.6, "How to Display Properties and Parameters in a Custom Property Panel."

21.3.2 What Happens at Runtime

If you associated the custom property panel with a specific component or task flow, at runtime the panel renders as a tab in the Component Properties dialog invoked from the specified component or task flow. If you did not associate the custom property panel with a specific component or task flow, at runtime the custom property panel renders as a tab in both the Page Properties and Component Properties dialogs for all pages, components, and task flows.

Note: In the Component Properties and Page Properties dialogs, custom panels are sized by the tab component containing the task flows. The size of the tab component itself is determined by a rule in the currently-applied skin. This rule is called `af|panelTabbed.Tab`.

21.3.3 How to Override Default Property Panels

The Component Properties dialog in Composer displays the following panels by default: Display Options, Child Components, Events, Style, Content Style, and Parameters. You can override a particular panel by using the default panel's name when you declare and register the custom panel in the `pe_ext.xml` file. For information about the default panels, see Section B.3, "Composer Default Add-Ons and Property Panels."

Example 21–10 shows the code to override the Display Options panel with a panel defined using a task flow called `custom-panel-task-flow`.

Example 21–10 Sample Code to Override a Default Panel

```
<addon-config>
  <panels>
    <panel name="oracle.adf.pageeditor.pane.generic-property-inspector"
title="Sample Panel"

icon="http://myforums.oracle.com/jive3/images/question-pts-available-16x16.gif"
```

```
taskflow-id="/WEB-INF/custom-panel-task-flow.xml#custom-panel-task-flow" />
  </panels>
  <property-panels>
    <property-panel name="samplepanel">
      <panel name="oracle.adf.pageeditor.pane.generic-property-inspector"
rendered="true"/>
    </property-panel>
  </property-panels>
</addon-config>
```

21.3.4 How to Exclude Default Property Panels

Out-of-the-box, all default property panels are displayed along with custom property panels that you configure. To hide a default property panel in the Page Properties or Component Properties dialog, you can add a line specifically for that panel and and set the `rendered` attribute to `false`.

Example 21–11 shows the `rendered` attribute set to `false` for the Content Style property tab of a `Command Button` component. For information about default property panels, see Section B.3, "Composer Default Add-Ons and Property Panels."

Example 21–11 rendered Attribute for a Property Panel

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <addon-config>
    <property-panels>
      <property-panel name="cmdbtn">
        <component>oracle.rich.CommandButton</component>
        <panel name="prop.panel.cmdbtn" />
        <panel name="oracle.adf.pageeditor.pane.content-style-editor"
rendered="false" />
      </property-panel>
    </property-panels>
  </addon-config>
  . . .
</pe-extension>
```

21.3.5 How to Selectively Render Property Panels

Custom property panels registered in the application are rendered in the Component Properties dialog when users click the **Edit** icon on a specified component. You can configure your application to render a property panel selectively based on different criteria like the role of a logged-in user, the page being viewed, and so on. To display property panels selectively, you can use an EL value in the `property-panel`'s `rendered` attribute as shown in the following example:

```
<property-panel name="global-but-just" rendered="#{bean.showProperty}">
```

21.3.6 How to Display Properties and Parameters in a Custom Property Panel

If you are building a custom property panel for a component, you can use `Show Property` components, available from the Composer tag library, to expose the component's attributes on the custom panel. In the page fragment defining the custom panel task flow, you must add one `Show Property` component for each attribute that you want to expose. `Show Property` is a declarative component that reads a property value on the component or task flow (for which you have created the custom panel), and displays an input field for that property in the custom panel. Users can edit property values in the panel using the input fields.

For more information about custom panels, see Section 21.3.1.1, "Creating a Custom Property Panel."

To add a `Show Property` component:

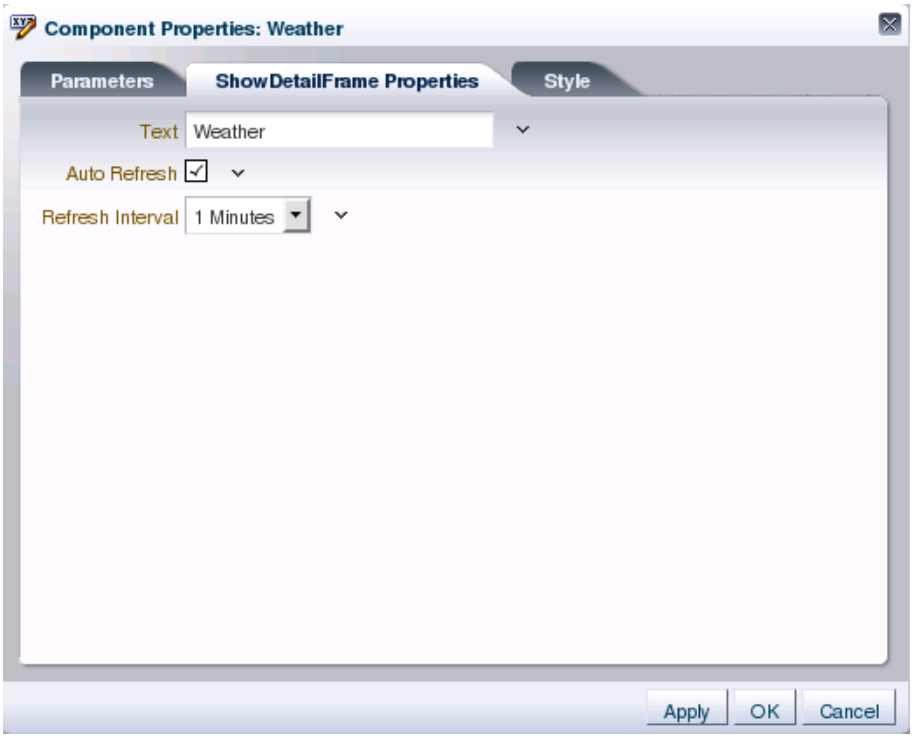
1. Open the page fragment that you created for the custom panel task flow.
2. From the Component Palette, select **Composer**.
3. Drag and drop a `Show Property` component for each component attribute that you want to expose in the custom panel.
4. To ensure that the attribute displays correctly in the custom panel, set the attributes on each `Show Property` component by referring to Section B.1.7, "Show Property."

> **Note:** You can use the `Show Property` component even on pages outside of Composer.

Example

Consider an example of a weather widget that is included inside a `Show Detail Frame` component. A custom property panel is created to enable users to edit the Text, Auto Refresh, and Refresh Interval attributes on the weather widget. It includes three fields for users to provide values for these attributes. To enable users to set the three attributes on the weather widget, one `Show Property` component is added for each attribute that is exposed in the panel. When a user edits the weather widget at runtime, the Component Properties dialog displays the custom panel with the three attributes, as shown in Figure 21–7.

Figure 21–7 Custom Panel to Display Show Detail Frame Component Properties



The weather widget is consumed in a page along with a `Poll` component as shown in the following example:

```
<cust:showDetailFrame text="Weather" id="sdf1" contentStyle="height:220.0px;"
                      partialTriggers="p1">
  <af:region value="#{bindings.weathertaskflowdefinition1.regionModel}"
             id="r1"/>
  <af:poll id="p1" interval="60000"/>
</cust:showDetailFrame>
```

The `Poll` component is used to deliver refreshed content to the weather widget at fixed intervals. The default interval is 60000 milliseconds.

The custom panel contains three `panelLabelandMessage` components to provide input fields for the `Text`, `Auto Refresh`, and `Refresh Interval` attributes. The fields on this panel are implemented as follows:

- A `showProperty` component is added to each `panelLabelandMessage` component as shown in the following example:

  ```
  <!-- Input field for setting the Text attribute, which displays on the
  component header -->
  <af:panelLabelAndMessage label="Text" id="plam2">
    <pedc:showProperty label="Text" id="sp1" simple="true"
                       value="#{sdfCustomPanel.text}"/>
  </af:panelLabelAndMessage>

  <!-- Checkbox to enable auto refresh, which controls whether the widget must be
  refreshed -->
  <af:panelLabelAndMessage label="Auto Refresh" id="plam3">
    <pedc:showProperty label="Auto Refresh" id="sp2" simple="true"
                       type="boolean" autoSubmit="true"
                       value="#{sdfCustomPanel.autoRefresh}"/>
  </af:panelLabelAndMessage>

  <!-- LOV to display values for Refresh Interval when auto refresh is enabled
  -->
  <af:panelLabelAndMessage label="Refresh Interval" id="plam4">
    <pedc:showProperty label="Refresh Interval" id="sp3" simple="true"
                       type="list" disabled="#{!sdfCustomPanel.autoRefresh}"
                       value="#{sdfCustomPanel.refreshInterval}"
                       selectItems="#{sdfCustomPanel.selectItems}"

  binding="#{sdfCustomPanelRequestBean.refreshIntervalShowProp}"/>
  </af:panelLabelAndMessage>
  ```

  where `sdfCustomPanel` references a managed bean, `SDFCustomPanelBean.java`.

- The task flow definition of the custom panel contains a reference to the bean:

  ```
  <managed-bean id="mb1">
    <managed-bean-name id="mb1_name">sdfCustomPanel</managed-bean-name>
    <managed-bean-class id="mb1_
  class">portal.weather.panel.SDFCustomPanelBean</managed-bean-class>
    <managed-bean-scope id="mb1scope">session</managed-bean-scope>
  </managed-bean>
  ```

- The `SDFCustomPanelBean.java` bean provides the logic to handle values for the different attributes exposed in the panel. In the bean, the `SelectionContext` API is used to read and write values for each attribute as follows:

- Get hold of the selected component, that is, the component on which the properties are displayed; in this example, a `Show Detail Frame` component, as the weather widget is included inside one.
- Read the attribute value on the selected component.
- Get the attribute value that the user specifies and set that value for the component attribute.

For example, the `Refresh Interval` attribute is enabled only if `Auto Refresh` in selected, and it displays a dropdown menu with the values defined in the managed bean. The `Poll` component on the page is wired to the `Show Property` component so that the widget refresh is driven by the value of the Show Property component. The code to handle the value for the `Refresh Interval` property is as follows:

```
. . .
/** Set the refresh interval attribute value on the component
public void setRefreshInterval(String _refreshInterval)
{
  this._refreshInterval = _refreshInterval;
}
**/

public String getRefreshInterval()
{
  if (_autoRefresh == true && _refreshInterval != null)
    return _refreshInterval;

  // Get hold of the component whose properties you want to edit
  SelectionContext selCtx =
    Context.getCurrentInstance().getSelectionContext();
  UIComponent selComp = selCtx.getSelectedComponent().getUIComponent();
  if (selComp instanceof ShowDetailFrame)
  {
    // Get the refresh interval value specified for the Poll component
    RichPoll pollComp = (RichPoll) selComp.findComponent("p1");
    if (_autoRefresh == false)
      _refreshInterval = "Refresh interval not applicable";
    else if (pollComp.getInterval() < 0)
      _refreshInterval = "60000";
    else
      _refreshInterval = Integer.toString(pollComp.getInterval());
  }

  return _refreshInterval;
}
. . .
public List<SelectItem> getSelectItems()
{
  return _selectItems;
}
. . .
private static List<SelectItem> _selectItems;
static
{
  _selectItems = new ArrayList<SelectItem>();
  SelectItem si = new SelectItem("60000", "1 Minutes");
  _selectItems.add(si);
  si = new SelectItem("120000", "2 Minutes");
  _selectItems.add(si);
```

```
    si = new SelectItem("180000", "3 Minutes");
    _selectItems.add(si);
  }
```

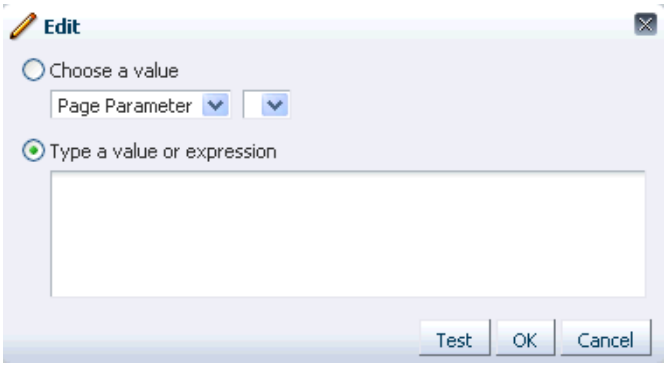
- When a user selects a value for the `Refresh Interval` property and clicks Apply in the Component Properties dialog, the save listener is called and the `save` event handler ensures that the value is saved back to the bean. The relevant code in the save event handler implementation is as follows:

```
public void save()
{
  . . .
  if (_autoRefresh)
  {
    aa = new AttributeComponentChange("interval", Integer.parseInt(_
refreshInterval));
    changeManager.addComponentChange(context, pollComp, aa);
    aa.changeComponent(pollComp);
  }
  else
  {
    aa = new AttributeComponentChange("interval", new Integer(-1));
    changeManager.addComponentChange(context, pollComp, aa);
    aa.changeComponent(pollComp);
  }
}
```

21.4 Extending the Expression Builder

All properties on the Parameters and Display Options tabs in the Component Properties dialog can take Expression Language (EL) expressions. The Expression Builder option available on such parameters opens the Edit dialog, which is a simple expression builder as shown in Figure 21–8. The expression builder is particularly useful when you want a value that is retrievable but otherwise unknown, for example, when you want a value to be the name of the current user or the current application skin.

Figure 21–8 Expression Language Editor



You can customize the expression builder to provide more options by making the necessary entries in the Composer extension file, `pe_ext.xml`. You can also configure Composer to disable EL or protect existing EL from overwrite.

This section explains how to extend the expression builder to add custom capabilities. It contains the following subsections:

- Section 21.4.1, "How to Extend the Expression Builder"
- Section 21.4.2, "How to Protect Expression Language"
- Section 21.4.3, "What Happens at Runtime"

21.4.1 How to Extend the Expression Builder

You can extend the expression builder by adding custom options to the expression value dropdown list. Custom options must be defined and included in the `pe_ext.xml` file. For each new option you want to add, you must define and include a `<selector>` element in the `pe_ext.xml` file.

To configure custom options in the Composer extension file:

1. Create a Java bean, for example `UserInformation.java`, with the logic to populate the expression dialog with your custom options. Example 21–12 shows the code of a sample Java bean used to display a new string, `User Name`, as a custom value in the expression dialog.

Example 21–12 Sample Code to Add a Custom Value in the Expression Dialog

```
package view;

import java.util.ArrayList;
import java.util.List;

import oracle.adf.view.page.editor.elbuilder.ELParameter;
import oracle.adf.view.page.editor.elbuilder.ELParameterValue;

public class UserInformation implements ELParameter {
  public String getName() {
    return mName;
  }

  public List<ELParameterValue> getValues() {
    if (mValues == null)
      {
        mValues = new ArrayList<ELParameterValue>();
        for(String[] args : mValueList)
          {
            mValues.add(new ELParameterValue(args[0], args[1]));
          }
      }
       return mValues;
  }
  private static String[][] mValueList =
    {new String[]{"User Name", "#{securityContext.userName}"}};

  private String mName = "User Name";
  private List<ELParameterValue> mValues;  }
```

2. Add an `<elbuilder-config>` element in the `pe_ext.xml` file to register the bean.

   For information about creating the extension file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

3. Add a `<selector>` element within the `<elbuilder-config>` section, and provide details about the bean you created, as shown in the following example:

```
<elbuilder-config>
```

```
  <!-- define selector -->
  <selector id="UserInformation">
    view.UserInformation
  </selector>

  <!-- include selector -->
  <selectors>
    <selector id="UserInformation" rendered="true"/>
  </selectors>
</elbuilder-config>
```

You can have multiple `<selector>` entries for different custom options.

Custom options display for all attributes by default. You can use the `rendered` attribute on `<selector>` to display an option conditionally, based on specific criteria.

4. Save the `pe_ext.xml` file.

21.4.2 How to Protect Expression Language

You can configure Composer to disable EL or protect existing EL from overwrite within an application or a specific .jspx page. This is especially useful when exposing elements to end users in Design view as described in Section 21.11, "Enabling Direct Select in Design View."

To define settings for an application, modify the following elements in adf-config.xml within the `<pe:page-editor-config>` tag:

- `<pe:allow-el>` defines whether or not EL can be run in application pages. The default is true.
- `<pe:protect-el>` defines whether or not EL in form elements is read-only. The default is false.

To define behavior for a specific .jspx page, use the following elements within the `<pe:pageCustomizable>` tag:

- `allowEL` (boolean) defines whether or not EL can be run in application pages. The default is true.
- `protectEL` (boolean) defines whether or not EL in form elements is read-only. The default is false.

The settings defined in the .jspx page override any settings defined in adf-config.xml.

21.4.3 What Happens at Runtime

The selector in the runtime expression builder displays the custom value along with the default values, as shown in Figure 21–9.

Figure 21–9 Expression Builder Displaying a Custom Value



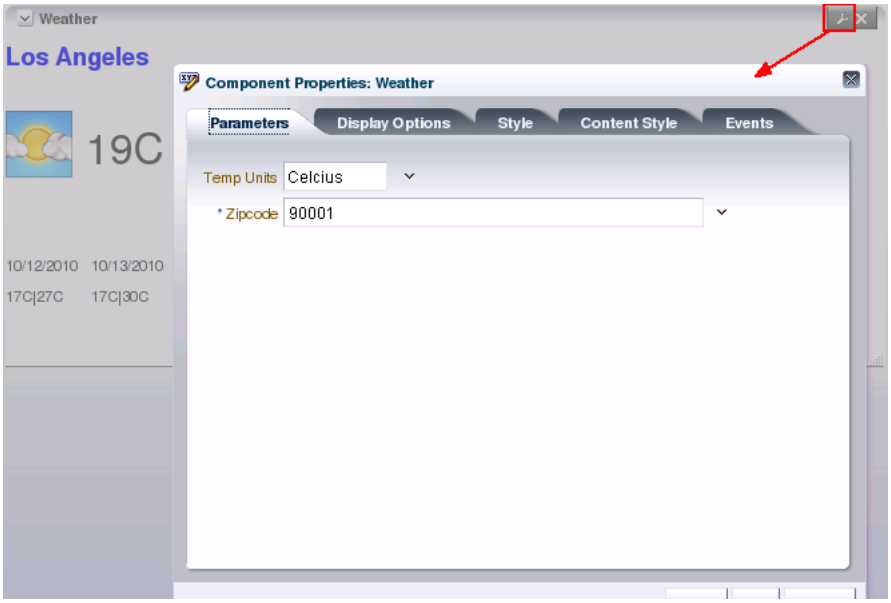
If EL is disabled, the expression builder will not be accessible, and EL will not be allowed in property fields.

If EL is protected, any EL already present as an attribute value cannot be changed in either the Parameters or Display Options tabs. (EL picked from a LOV is allowed.)

21.5 Configuring Custom LOVs or Pickers for Task Flow Parameters

Typically, while editing a task flow parameter in Composer, the user enters a value in the text box provided in the Component Properties dialog, as shown in Figure 21–10. However, many a time the user is confused about what value to specify in a particular field. Rather than leave them guessing, as a developer who knows the values that each parameter can take, you can make it easier for users by listing all valid options, wherever possible. You can configure Composer to display a list of values (LOV) or a picker against any parameter.

Figure 21–10 Parameters Tab on the Component Properties Dialog for a Task Flow



You can display a predefined set of values for a parameter using the following options:

- Static LOV

  To display a list of predefined values.

- Dynamic LOV

  To display a list of values generated by evaluating an EL value that is computed when the page is run.

- Global LOV

  To display a global list of values that can be used in any task flow in the application.

- Picker

  To display all values in a picker format, for example, a document picker.

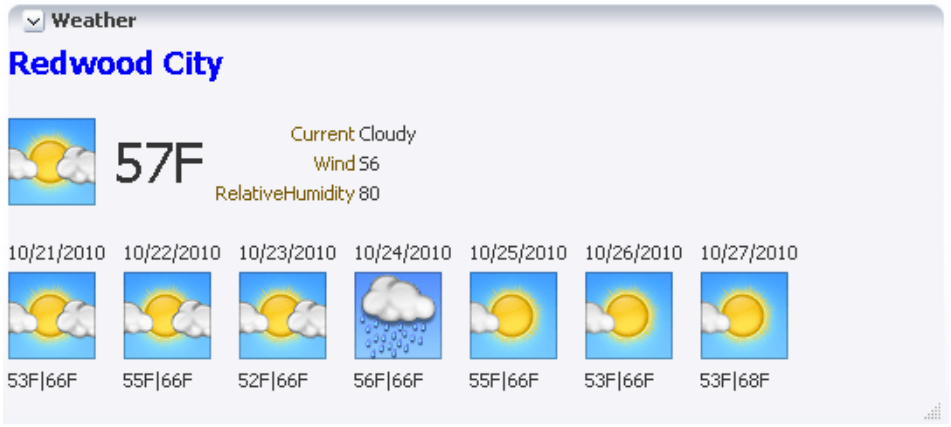
This section describes how to configure different types of LOVs for task flow parameters. It contains the following subsections:

- Section 21.5.1, "How to Configure an LOV"
- Section 21.5.2, "What Happens at Runtime"

Example Used to Describe the Procedures in This Section

Some of the procedures in this section use the example of a simple weather task flow, `weather-task-flow-definition`, which displays weather information for a selected city, as shown in Figure 21–11.

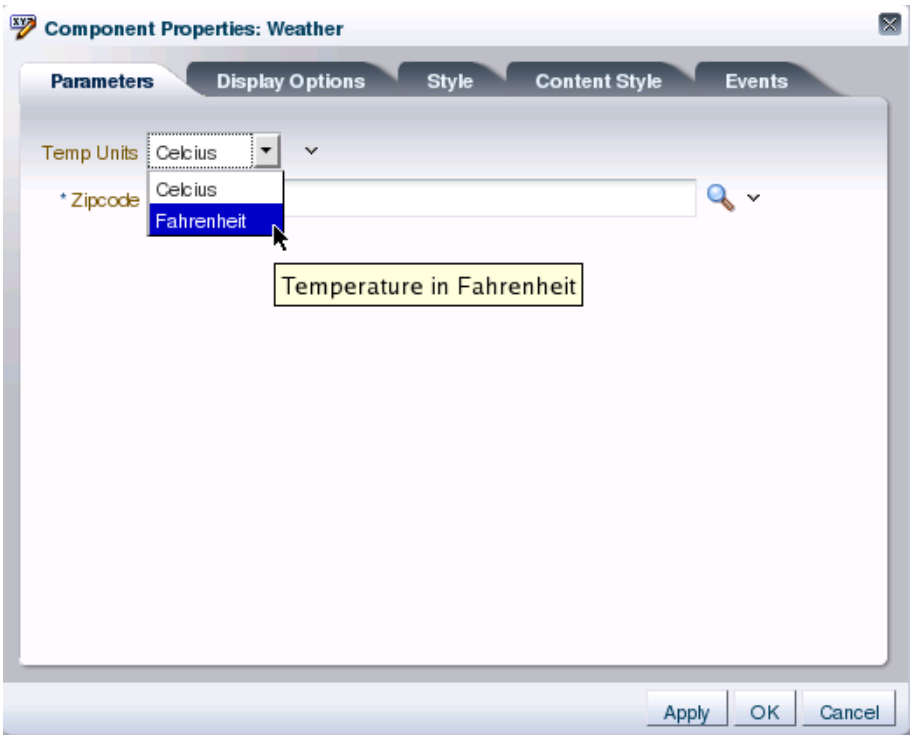
Figure 21–11 Weather Task Flow



The task flow provides two parameters, `zipCode` and `tempUnits`, that users can modify to display weather details for a selected city and in a selected temperature unit (Fahrenheit or Celsius) respectively. This weather task flow example is used to explain how to configure the following types of LOVs:

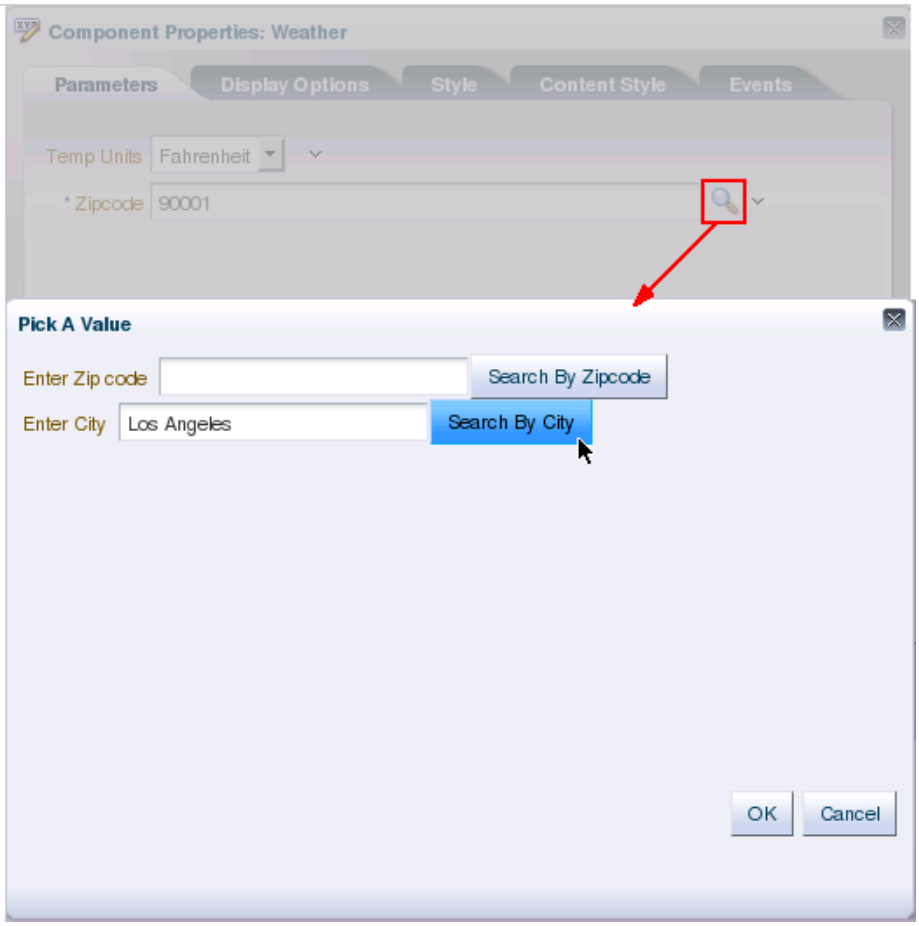
- A static LOV with the values `Celsius` and `Fahrenheit` for the `tempUnits` parameter, as shown in Figure 21–12.

Figure 21–12 Static LOV for the tempUnits Parameter



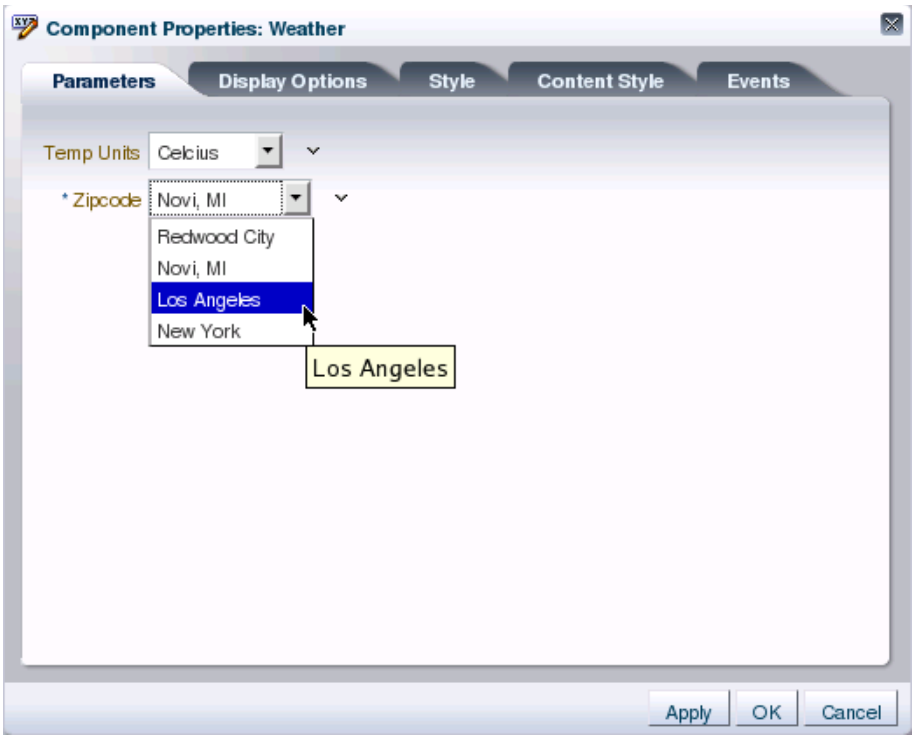
- A picker with two fields that enable users to specify a city name or ZIP code (Figure 21–13) to view corresponding details in the weather task flow.

Figure 21–13 City Picker for the zipCode Parameter



- Alternatively, a dynamic LOV for the `zipCode` parameter to display values (Figure 21–14) based on an EL value that is evaluated at runtime.

Figure 21–14 Dynamic LOV for the zipCode Parameter



The weather task flow includes a task flow definition file, `weather-task-flow-definition.xml`, with the code shown in Example 21–13.

Example 21–13 The weather-task-flow-definition.xml FIle

```
<?xml version="1.0" encoding="US-ASCII" ?>
<adfc-config xmlns="http://xmlns.oracle.com/adf/controller" version="1.2">
  <task-flow-definition id="weather-task-flow-definition">
    <default-activity id="defaultactivity1">weather</default-activity>
    <input-parameter-definition id="inputparam1">
      <name id="inputparam1name">zipcode</name>
      <value>#{pageFlowScope.zipcode}</value>
      <class>java.lang.String</class>
      <required/>
    </input-parameter-definition>
    <input-parameter-definition id="inputparam2">
      <name id="ip2name">tempUnits</name>
      <value>#{pageFlowScope.tempUnits}</value>
      <class>java.lang.String</class>
    </input-parameter-definition>
    <view id="weather">
      <page>/weather.jsff</page>
    </view>
    <use-page-fragments/>
  </task-flow-definition>
</adfc-config>
```

The parameters, shown in **bold** text, are defined in such a way that values provided by users at runtime are stored in a session bean, `pageFlowScope`, from where they can be read by other components.

When you edit the task flow at runtime, the Parameters tab displays two input text fields to provide parameter values. The following sections explain how to enable LOVs for these task flow parameters so that users can choose from a list of predefined of options.

21.5.1 How to Configure an LOV

This section describes how to configure different types of LOVs for task flow parameters. It includes the following subsections:

- Section 21.5.1.1, "Configuring a Static LOV"
- Section 21.5.1.2, "Configuring a Dynamic LOV"
- Section 21.5.1.3, "Configuring a Picker"
- Section 21.5.1.4, "Configuring an LOV from a Global List"

21.5.1.1 Configuring a Static LOV

Use a static LOV to display a list of predefined values against a parameter. You can configure a static LOV by defining an `<lov-config>` section with nested `<enumeration>` elements in the Composer extension file, `pe_ext.xml`. This section uses the weather task flow example and describes how to configure a static LOV for the `tempUnits` parameter.

To configure a static LOV:

1. If it does not already exist, create the Composer extension file, `pe_ext.xml` in the `META-INF` directory under the project's Web context root (for example, in the *`APPLICATION_HOME`*`\Portal\adfmsrc\META-INF` directory):

   a. From the **File** menu, select **New**.

   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.

   c. Click **OK**.

   Name the file `pe_ext.xml`.

2. Add an `<lov-config>` section and include the task flow definition with nested `<enumeration>` elements, as shown in the following code example:

```
<lov-config>
  <!-- Entry for the task flow in which you are configuring the LOV -->
  <task-flow-definition
taskflow-id="/WEB-INF/weather-task-flow-definition.xml#weather-task-flow-defini
tion">

    <!-- Specifying the parameter for which you want to define a static LOV -->
    <input-parameter-definition>
      <name>tempUnits</name>
      <!-- List of values to be displayed for the parameter -->
      <enumeration inline="true">
        <item>
          <name>Celcius</name>
          <value>c</value>
          <description>Temperature in Celcius</description>
        </item>
        <item>
          <name>Fahrenheit</name>
          <value>f</value>
```

```
          <description>Temperature in Fahrenheit</description>
        </item>
      </enumeration>
    </input-parameter-definition>

  </task-flow-definition>
</lov-config>
```

You can configure any number of static LOVs within the `<lov-config>` section.

> **Note:** To allow translatable strings to be provided for a parameter name, you can use an EL value for the name, for example, `#{resourceBundle.temp_units}`.

3. Save the file.

21.5.1.2 Configuring a Dynamic LOV

Use a dynamic LOV to display a list that is generated dynamically at runtime by evaluating an EL value. This section uses the weather task flow example and describes how to configure a dynamic LOV for the `zipCode` parameter. To keep the example simple, the managed bean only provides static values that are displayed on the `Zipcode` field at runtime.

To configure a dynamic LOV:

1. Create a bean, for example, `weatherBean`, and define the list of values for the `Zipcode` parameter. The following example shows the relevant code snippet:

```
package portal.weather;

  . . .

  public List<SelectItem> getCitiesList() {
    return citiesList;
  }

  private static final String F = "&deg;F";
  private static final String C = "&deg;C";
  private static final String [][] selectItems = {{"Redwood City", "94065"},
                                                  {"Novi, MI", "48375"},
                                                  {"Los Angeles", "90001"},
                                                  {"New York", "10036"}};

  private static final List<SelectItem> citiesList = new
ArrayList<SelectItem>();
    static {
      for (String[] city : selectItems){
        citiesList.add(new SelectItem(city[1], city[0]));
    }
  }
```

2. Register `weatherBean` in the application's `faces-config.xml` file, as shown in the following example:

```
<managed-bean>
  <managed-bean-name>weatherBean</managed-bean-name>
  <managed-bean-class>portal.weather.WeatherBean</managed-bean-class>
  <managed-bean-scope>request</managed-bean-scope>
```

```
</managed-bean>
```

3. If it does not already exist, create the Composer extension file, `pe_ext.xml` in the `META-INF` directory under the project's Web context root (for example, in the *APPLICATION_HOME*`\Portal\adfmsrc\META-INF` directory):

   a. From the **File** menu, select **New**.

   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.

   c. Click **OK**.

   Name the file `pe_ext.xml`.

4. Add an `<lov-config>` section, include the task flow definition with an `<enumeration>` element, and reference the managed bean for the parameter value, as shown in the following example:

```
<lov-config>
  <task-flow-definition
taskflow-id="/WEB-INF/weather-task-flow-definition.xml#weather-task-flow-defini
tion">
    <input-parameter-definition>
      <name>zipcode</name>
      <value>90001</value>
      <enumeration items="#{weatherBean.citiesList}"/>
    <input-parameter-definition>
  </task-flow-definition>
</lov-config>
```

   You can configure any number of dynamic LOVs within the `<lov-config>` section.

5. Save the file.

21.5.1.3 Configuring a Picker

Typically, a picker enables users to select a value from a popup dialog that displays all available options, for example, a color picker or date picker. Configuring a picker for a task flow parameter involves the following high level tasks:

- Create the picker task flow with a UI of your choice.
- Define parameters on the picker task flow. These parameters are used to populate the main task flow parameter (the one for which you are configuring the picker).
- Configure the main task flow parameter to display the picker that you created. You must add this configuration in the `pe_ext.xml` file.

This section explains these steps in detail using the weather task flow as an example. It shows how to configure a city picker for the `zipCode` parameter. The picker task flow enables users to specify the city name and get the ZIP code for that place. The ZIP code from the picker is then pushed to the Zipcode field in the weather task flow. This section includes the following subsections:

- Section 21.5.1.3.1, "Creating the Picker Task Flow"
- Section 21.5.1.3.2, "Registering the Picker for a Task Flow Parameter"

21.5.1.3.1 Creating the Picker Task Flow

Create a picker task flow with a JSFF fragment. Design the fragment to display fields or options of your choice.

To create the picker task flow:

1. In your application project, create an ADF task flow, for example, `weather-picker-task-flow-definition`:
   a. From the **File** menu, choose **New**.
   b. In the New Gallery dialog, expand **Web Tier**, select **JSF**, then **ADF Task Flow**.
   c. Click **OK**.
2. Add a `view` object to the task flow, for example, `weatherPicker`.
3. Double-click the view to create the `weatherPicker.jsff` fragment.

   See Also: "Getting Started with ADF Task Flows" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for information about creating task flows.

4. Design the fragment with components of your choice. Example 21–14 shows a sample task flow containing two `Input Text` components that enable users to search for ZIP codes in an XML file, and two UI controls, `OK` and `Cancel`.

Example 21–14 Sample code in the Picker JSFF Fragment

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:f="http://java.sun.com/jsf/core">
  <af:panelStretchLayout id="psl1">
    <f:facet name="center">
      <af:panelGroupLayout id="pgl1" layout="vertical">

        <af:panelGroupLayout id="pgl2" layout="horizontal">
          <!-- Text field and Search button to search for a city by specifying the
ZIP code -->
          <af:inputText label="Enter Zip Code" id="it1"
                        value="#{pageFlowScope.weatherPicker.searchZipcode}"/>
          <af:commandButton text="Search By Zipcode" id="cb1"

actionListener="#{pageFlowScope.weatherPicker.zipcodeSearch}"/>
        </af:panelGroupLayout>

        <af:panelGroupLayout id="pgl3" layout="horizontal">
          <!-- Text field and Search button to search for a city by specifying the
city name -->
          <af:inputText label="Enter City" id="it2"
                        value="#{pageFlowScope.weatherPicker.searchCity}"/>
          <af:commandButton text="Search By City" id="cb2"

actionListener="#{pageFlowScope.weatherPicker.citySearch}"/>
        </af:panelGroupLayout>

        <!-- Output components to display the result from the search -->
        <af:outputFormatted value="#{pageFlowScope.weatherPicker.city}"
                            id="of2"/>
        <af:outputFormatted value="#{pageFlowScope.weatherPicker.zipcode}"
                            id="of3"/>

      </af:panelGroupLayout>
    </f:facet>
    <f:facet name="bottom">
      <af:panelGroupLayout layout="horizontal" halign="end" id="pkrPgl">
```

```
        <!-- OK button -->
        <af:commandButton action="ok"

actionListener="#{pageFlowScope.weatherPicker.populateReturnValue}"
                          text="#{Bundle.OK_BUTTON_TEXT}" id="okBtn"/>
        <af:spacer width="5" id="s1"/>

        <!-- Cancel button -->
        <af:commandButton
actionListener="#{pageFlowScope.weatherPicker.closePickerPopup}"
                          text="#{Bundle.CANCEL_BUTTON_TEXT}"
                          id="cnclBtn"/>
      </af:panelGroupLayout>
    </f:facet>
  </af:panelStretchLayout>
</jsp:root>
```

The action listeners on the `OK` and `Cancel` buttons reference a managed bean, `weatherPicker`. This bean also contains the logic to populate the parameter on the main task flow with the value specified in the picker.

5. Create the bean that you referenced in the picker task flow definition and add the logic to return values to the main task flow.

   The following example shows the relevant code snippet of the `weatherPicker` bean, which contains the logic to search for a city by name and return the ZIP code:

```
package portal.weather.picker;

. . .

  public void populateReturnValue(ActionEvent ae) {
    Map map =
(Map)RequestContext.getCurrentInstance().getPageFlowScope().get("outParam");
    if (map != null)
      map.put("outParamVal", _zipcode);
  }

  public void closePickerPopup(ActionEvent ae) {
    UIComponent component = ae.getComponent();
      while (!(component instanceof RichPopup))
        component = component.getParent();
        if (component != null)
          ((RichPopup)component).cancel();
  }
  . . .
```

6. Register the `weatherPicker` bean in the task flow definition file, as shown in the following example:

```
<managed-bean id="pickerbean">
  <managed-bean-name id="pickerbeanname">weatherPicker</managed-bean-name>
  <managed-bean-class
id="pickerbeanclass">portal.weather.picker.WeatherPicker</managed-bean-class>
  <managed-bean-scope id="pickerbeanscope">pageFlow</managed-bean-scope>
</managed-bean>
```

7. Define two input parameters, `inParamVal` and `outParam`, on the picker task flow, as shown in the following example:

```
<!-- Parameter used to store the value that must be passed to the main task
flow -->
<input-parameter-definition id="pickerip1">
  <name id="pickerip1name">outParam</name>
  <value>#{pageFlowScope.outParam}</value>
  <class>java.util.Map</class>
</input-parameter-definition>

<!-- Parameter used to store the value specified by the user in the picker -->
<input-parameter-definition id="pickerip2">
  <name id="pickerip2name">inParamVal</name>
  <value>#{pageFlowScope.inParamVal}</value>
  <class>java.lang.String</class>
</input-parameter-definition>
```

These parameters are referenced in `pageFlowScope` and will be used to read and populate the main task flow parameter, `zipCode`. When a user clicks the picker icon next to the Zip Code field, the current value of the field is passed to the session bean using `inParamVal`. When a user specifies a city name in the picker task flow and clicks OK, the bean gets the ZIP code for that city and passes it to the main weather task flow using `outParam`.

8. Define task flow return activities and control flow rules for the `OK` and `Cancel` buttons. These activities and rules ensure that control is returned to the task flow when either of the buttons is clicked. You can define task flow return activities and control flow rules as shown in the following example:

```
<!-- The outcome to be returned to the task flow if a user clicks OK -->
<task-flow-return id="ok">
  <outcome id="ok1">
    <name id="okayed">ok</name>
  </outcome>
</task-flow-return>
<!-- The outcome to be returned to the task flow if a user clicks Cancel -->
<task-flow-return id="cancel">
  <outcome id="cancel1">
    <name id="canceled">cancel</name>
  </outcome>
</task-flow-return>

<!-- Rules to pass control flow to the ok activity if the user clicks OK and to
the cancel activity if the user clicks Cancel -->
<control-flow-rule id="flowrule1">
  <from-activity-id id="fromactivity1">weatherPicker</from-activity-id>
  <control-flow-case id="flowcase1">
    <from-outcome id="fromoutcome1">ok</from-outcome>
    <to-activity-id id="toactivity1">ok</to-activity-id>
  </control-flow-case>
  <control-flow-case id="flowcase2">
    <from-outcome id="fromoutcome2">cancel</from-outcome>
    <to-activity-id id="toactivity2">cancel</to-activity-id>
  </control-flow-case>
</control-flow-rule>
```

The task flow definition will now have code similar to the following example:

```
<?xml version="1.0" encoding="US-ASCII" ?>
<adfc-config xmlns="http://xmlns.oracle.com/adf/controller" version="1.2">
  <task-flow-definition id="weather-picker-task-flow-definition">
    <default-activity id="pickeractivity1">weatherPicker</default-activity>
    <input-parameter-definition id="pickerip1">
```

```
      <name id="pickerip1name">outParam</name>
      <value>#{pageFlowScope.outParam}</value>
      <class>java.util.Map</class>
    </input-parameter-definition>
    <input-parameter-definition id="pickerip2">
      <name id="pickerip2name">inParamVal</name>
      <value>#{pageFlowScope.inParamVal}</value>
      <class>java.lang.String</class>
    </input-parameter-definition>
    <managed-bean id="pickerbean">
      <managed-bean-name id="pickerbeanname">weatherPicker</managed-bean-name>
      <managed-bean-class
id="pickerbeanclass">portal.weather.picker.WeatherPicker</managed-bean-class>
      <managed-bean-scope id="pickerbeanscope">pageFlow</managed-bean-scope>
    </managed-bean>
    <view id="weatherPicker">
      <page>/weatherPicker.jsff</page>
    </view>
    <task-flow-return id="ok">
      <outcome id="ok1">
        <name id="okayed">ok</name>
      </outcome>
    </task-flow-return>
    <task-flow-return id="cancel">
      <outcome id="cancel1">
        <name id="canceled">cancel</name>
      </outcome>
    </task-flow-return>
    <control-flow-rule id="flowrule1">
      <from-activity-id id="fromactivity1">weatherPicker</from-activity-id>
      <control-flow-case id="flowcase1">
        <from-outcome id="fromoutcome1">ok</from-outcome>
        <to-activity-id id="toactivity1">ok</to-activity-id>
      </control-flow-case>
      <control-flow-case id="flowcase2">
        <from-outcome id="fromoutcome2">cancel</from-outcome>
        <to-activity-id id="toactivity2">cancel</to-activity-id>
      </control-flow-case>
    </control-flow-rule>
    <use-page-fragments/>
  </task-flow-definition>
</adfc-config>
```

9. Save the task flow definition file.

21.5.1.3.2 Registering the Picker for a Task Flow Parameter To display a custom picker for a task flow parameter, you must configure the picker for that parameter in the Composer extension file, `pe_ext.xml`.

To configure a picker for a task flow parameter:

1. If it does not already exist, create the Composer extension file, `pe_ext.xml` in the `META-INF` directory under the project's Web context root (for example, in the *`APPLICATION_HOME`*`\Portal\adfmsrc\META-INF` directory):
   a. From the **File** menu, select **New**.
   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.
   c. Click **OK**.

Name the file `pe_ext.xml`.

2. Add an `<lov-config>` section and include the task flow definition with an `<enumeration>` element referencing the picker task flow, as shown in the following code example:

```
<?xml version="1.0" encoding="UTF-8"?>
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
  <lov-config>
    <task-flow-definition
taskflow-id="/WEB-INF/weather-task-flow-definition.xml#weather-task-flow-defini
tion">
      <!-- Parameter for which you defined the picker -->
      <input-parameter-definition>
        <name>zipCode</name>
        <value>null</value>
        <!-- enumeration items coming from the picker task flow -->
        <enumeration
picker="/WEB-INF/weather-picker-task-flow-definition.xml#weather-picker-task-fl
ow-definition"/>
      </input-parameter-definition>
    </task-flow-definition>
  </lov-config>
</pe-extension>
```

   You can configure pickers for any number of parameters within the `<lov-config>` section.

3. Save the file.

When you edit the main task flow at runtime, the Component Properties dialog displays a magnifier icon next to that parameter. Clicking the icon invokes the picker task flow, in which users can select a value.

21.5.1.4 Configuring an LOV from a Global List

If you want to reuse an LOV in different task flow parameters within the application, you can define the LOV at a global level and reference it from the task flow parameters.

To configure an LOV from a global list:

1. If it does not already exist, create the Composer extension file, `pe_ext.xml` in the `META-INF` directory under the project's Web context root (for example, in the *`APPLICATION_HOME`*`\Portal\adfmsrc\META-INF` directory):

   a. From the **File** menu, select **New**.

   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.

   c. Click **OK**.

   Name the file `pe_ext.xml`.

2. Add an `<lov-config>` section and list the values within an `<enumeration>` element (outside of the task flow definition), as shown in the following code example:

```
<lov-config>
<!-- Global enumerations ->
  <enumeration id="temperature">
    <item>
      <name>Celcius</name>
```

```
      <value>c</value>
      <description>Temperature in Celcius</description>
    </item>
    <item>
      <name>Fahrenheit</name>
      <value>f</value>
      <description>Temperature in Fahrenheit</description>
    </item>
  </enumeration>
  <task-flow-definition . . . >
  . . .
<lov-config>
```

You can configure any number of global LOVs within the `<lov-config>` section.

3. Reference this global LOV from any input parameter definition in the `pe_ext.xml` file, as shown in the following example:

```
<task-flow-definition
taskflow-id="/WEB-INF/weather-task-flow-definition.xml#weather-task-flow-definition">
  <input-parameter-definition>
    <name>tempUnits</name>
    <value>Celsius</value>
    <enumeration ref="temperature"/>
  </input-parameter-definition>
</task-flow-definition>
```

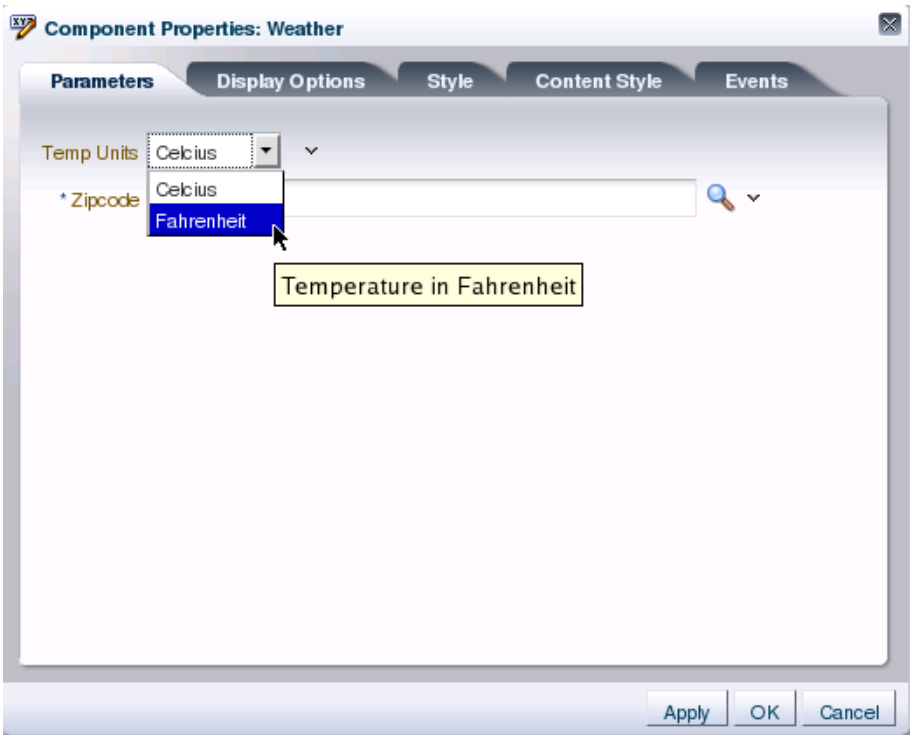
4. Save the file.

21.5.2 What Happens at Runtime

Depending on the type of LOV you configured for the task flow parameter, users will see different options to specify parameter values.

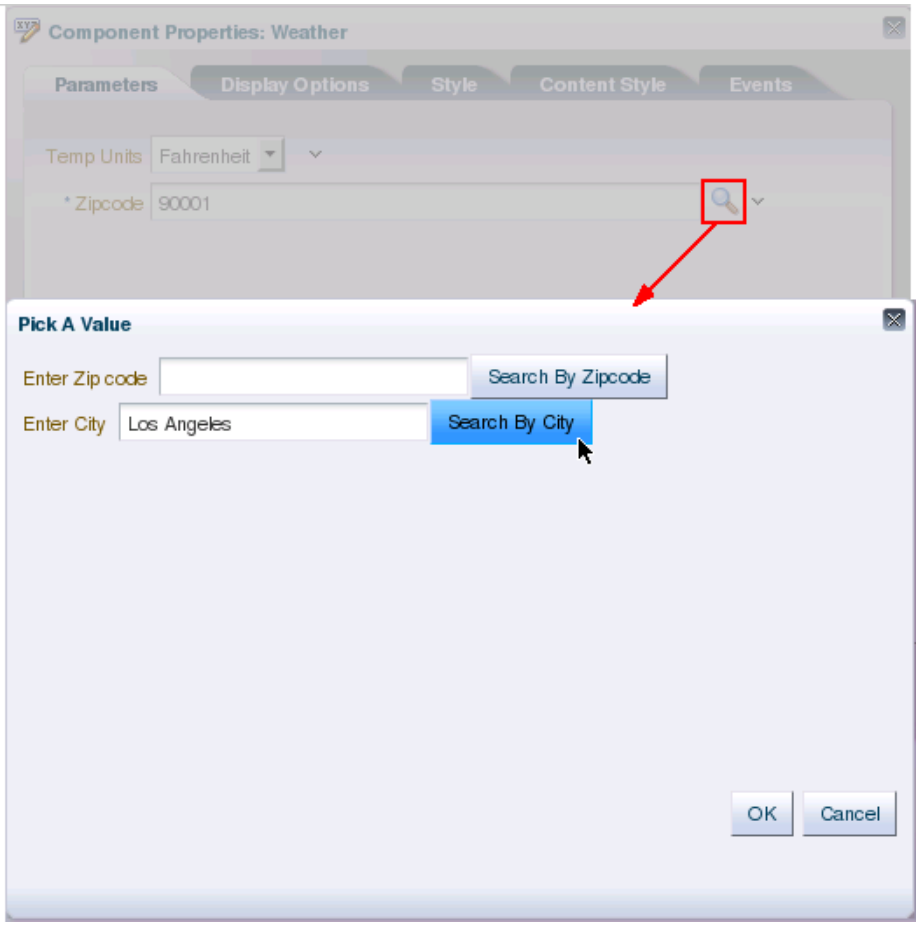
The Temp Units field, shown in Figure 21–15, lists values from the static LOV that you defined in the `pe_ext.xml` file.

Figure 21–15 Static LOV



If you configured a picker for the Zipcode field, the field shows a magnifier icon to its right. This icon invokes a picker, as shown in Figure 21–16, that allows users to select a city by providing a name or ZIP code.

Figure 21–16 Picker Task Flow Configured for the Zipcode Field



21.6 Configuring Event Handlers for Composer UI Events

Event handlers are Java classes registered with Composer and called when a users performs an action on the page. For example, when a user clicks a **Save** button, Composer calls back into the application code to give the application a chance to respond to the Save action—or *event*. In addition to the event handlers that Composer provides by default, you might want to configure the application to perform additional workflows. This is useful in performing application-specific tasks on such events. Moreover, this is the recommended approach to, for example, save the changes on a custom property panel.

Composer provides a means of registering event handlers for different events. This section describes how. It contains the following subsections:

- Section 21.6.1, "How to Create and Register Handlers for Composer UI Events"
- Section 21.6.2, "What Happens When You Create and Register Event Handlers"

21.6.1 How to Create and Register Handlers for Composer UI Events

When you register an event handler with Composer, it is called when the corresponding event is fired in the Composer UI. This section describes how to create an event handler and register it with Composer. It contains the following subsections:

- Section 21.6.1.1, "UI Events that Support Event Handler Registration"

- Section 21.6.1.2, "Creating a Save Event Handler: Example"
- Section 21.6.1.3, "Registering an Event Handler with Composer"

21.6.1.1 UI Events that Support Event Handler Registration

Table 21–1 lists the UI events for which Composer currently supports registering of handlers.

Table 21–1 Events for Which Registering Handlers are Supported

Event	Cause	Event Type	Listener Interface to Be Implemented	Method to Be Implemented	Event Parameters
Save	Invoked when a user clicks the **Save** or **Save and Label** button on the Composer toolbar, or the **Apply** or **OK** button in the Component Properties dialog or Page Properties dialog.	`save`	`oracle.adf.view.page.event.SaveListener`	`processSave`	`SaveEvent`
Close	Invoked when a user clicks the **Close** button on the Composer toolbar.	`close`	`oracle.adf.view.page.event.CloseListener`	`processClose`	`CloseEvent`
Deletion	Invoked when a user deletes the component.	`delete`	`oracle.adf.view.page.event.DeletionListener`	`processDeletion`	`DeletionEvent` Get the deleted component using `DeletionEvent`'s `getComponent` method.
Addition	Invoked when a user adds a component to the page from the Resource Catalog by clicking the **Add** button against an item.	`add`	`oracle.adf.view.page.event.AdditionListener`	`processAddition`	`AdditionEvent`
Selection	Invoked when a user selects: ■ The **Edit** icon on a Panel Customizable or Show Detail Frame component in Design view ■ A component on the page or in the hierarchy in Source view	`select`	`oracle.adf.view.page.event.SelectionListener`	`processSelection`	`SelectionEvent`
Attribute Modification	Invoked when a user changes an attribute.	`attribute-change`	`oracle.adf.view.page.editor.event.AttributeChangeListener`	`processAttributeChange`	`AttributeChangeEvent`

21.6.1.2 Creating a Save Event Handler: Example

To register an event with Composer, you must first create a Java class and implement the appropriate listener for the event handler. This section describes the steps to create a Save event handler. You can perform similar steps to create event handlers for all the supported events listed in Table 21–1.

A Save event handler is called when a user clicks the **Save** button on the Composer toolbar or the **Apply** or **OK** button in the Component Properties or Page Properties dialog. A Save event handler must implement `oracle.adf.view.page.editor.event.SaveListener`. The `isCommit` method of the Save event can be used to differentiate between changes made to the page and to component properties. A value of `true` implies that the user clicked the Save button on the Composer toolbar to save changes made to the page. A value of `false` implies that the user clicked the Apply or OK button in the Component Properties dialog to save changes made to component properties.

To create a Save event handler:

1. In JDeveloper, select the **Portal** project, and from the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.
3. In the Create Java Class dialog, specify a name for the class, for example, `SaveHandler`.
4. Under the Optional Attributes section, add the `oracle.adf.view.page.editor.event.SaveListener` interface.

   Note: If you are not able to see the `oracle.adf.view.page.editor` classpath, make sure that Composer is included in the project's libraries and classpath. To do this, right-click the project in the Application Navigator and select **Project Properties**. In the Project Properties dialog, select **Libraries and Classpath**, and add **Composer** to the list.

5. Click **OK**.

   The Java class source looks like the following:

```
package view;

import javax.faces.event.AbortProcessingException;

import oracle.adf.view.page.editor.event.SaveEvent;
import oracle.adf.view.page.editor.event.SaveListener;

public class SaveHandler implements SaveListener {
  public Class1() {
    super();
  }
  public void processSave(SaveEvent saveEvent) throws AbortProcessingException
{
// Your implementation goes here
  }
}
```

   You must declare the `processSave` method as `throws AbortProcessingException` because the method may throw this exception if

the event must be canceled. You can include the reason for canceling this event in the Exception object when you create it.

On throwing this exception, further processing of this event is canceled and the listeners that are in the queue are skipped.

You can create event handlers for all supported events by performing steps similar to these.

Tip: You can use the `SaveEvent.isCommit` method to

21.6.1.3 Registering an Event Handler with Composer

After creating and implementing an event handler, you must register it with Composer. Registration is necessary for ensuring that the handler is called back by Composer when the corresponding event occurs in the UI.

Register event handlers in the Composer extension file, `/META-INF/pe_ext.xml`. For more information about creating this file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

To register an event handler:

1. In the application's `pe_ext.xml` file, add the following entries:

```
<event-handlers>
  <event-handler event="save">view.SaveHandler</event-handler>
</event-handlers>
```

   The values you provide for the `event` attribute and between the `event-handler` tags are unique to the type of event being entered and the name you specified for the event class.

2. Save the file.

21.6.2 What Happens When You Create and Register Event Handlers

At runtime, registered event handlers are called according to the sequence in the extension file and according to the order in which they were found on the class path. Composer's native event handler is called last.

On invocation of an event handler's `process`*`EventName`* method, if an event handler throws `AbortProcessingException`, then the event is canceled and no further event handlers are called, including Composer's native event handlers.

If, however, an error occurs while instantiating an event handler, then Composer continues with the next event handler. A warning message is logged.

21.6.3 Performing Additional Composer Event Handler Configurations

For every UI event triggered on the page in Composer, the corresponding event handler calls back a method from a listener registered with the application. By performing some additional configurations, you can specify the sequence in which event handlers must be executed and configure a listener to terminate a process or notify Composer that the event has been handled.

Specify a Sequence Number for an Event Handler

By specifying the sequence for event handlers, you can decide on the order in which the event handlers, and therefore the listeners, are called. You can assign a sequence

number to a listener or modify the default value by defining a `sequence` attribute against the registered event handler in the `pe_ext.xml` file.

To specify a sequence number for a listener:

1. In the application's `pe_ext.xml` file, locate the `event-handler` element for the handler that you want to sequence.
2. Add a `sequence` attribute as follows:

```
<event-handlers>
  <event-handler event="save" sequence="101">view.SaveHandler</event-handler>
</event-handlers>
```

   The value for the `sequence` attribute must be a positive integer. If you do not define this attribute, the event handler is internally assigned a default sequence number of `100`.

   Composer built-in listeners and listeners with no sequence numbers are assigned a default sequence number of `100`. If you want your event handler to be called before other event handlers, you must specify a value lesser than `100`.

3. Save the file.

Stop Event Processing and Notify Composer

You can configure an event handler to terminate processing of the current event and all subsequent events by throwing an exception. For this, you must declare the method used while implementing the listener as `throws AbortProcessingException`. In addition, you can configure the handler to notify Composer that the event has been processed by using the `Event.setEventHandled(true)` method.

For example, you can configure a `delete` event handler to terminate the current event and all pending delete events and throw an exception when a user attempts to delete a component in Composer. To enable this, you must implement the `DeletionListener` interface as shown in Example 21–15.

Example 21–15 DeletionListener Implementation

```
public class DeleteHandler implements DeletionListener
{
  ...
  public void processDeletion(DeletionEvent delEvent) throws
AbortProcessingException
  {
    // Get the component that must be deleted
    UIComponent comp = delEvent.getComponent();

    if (comp != null)
    {
      try
      {
        // Assuming that a custom method, handleDelete(comp), handles deletion of
        // a component and returns true on successful deletion and false in case
        // of failure
        boolean deleteSucceeded = handleDelete(comp);

        // If deletion failed, then notify Composer that the delete event
        // has been handled. No further events are processed.
        if (!deleteSucceeded)
          delEvent.setEventHandled(true);
      }
```

```
      catch (Exception e)
      {
       // Catch Exception throw by handleDelete(comp) and handle it by throwing
       // AbortProcessingException to stop processing of delete events.
       throw new AbortProcessingException(e)
      }
    }
  }
  ...
}
```

21.7 Configuring Drop Handlers in the Resource Catalog

Drop handlers are Java classes registered with Composer and called when users click an Add link in the Resource Catalog. A drop handler declares the data flavors it can handle. Each resource in the catalog has a flavor. When a user clicks an Add link next to a resource, Composer queries all the registered drop handlers to check if they can handle the flavor. If only one drop handler can handle that flavor, then control is passed to that drop handler and the resource is added to the page immediately. If more than one drop handler can handle the flavor, then a context menu displays available drop handlers to users. The resource is dropped on the page when a user selects a Java drop handler from the context menu.

An example for a resource with multiple Java drop handlers is the Personal Documents subfolder available within the Documents folder, if you configured the Documents service in your catalog. This subfolder can be added as a Content Presenter, Document List Viewer, or Document Library.

> **Note:** The `AdditionListener` mechanism used to handle an `Add` operation in release 11.1.1.1 is still available, but now Composer calls drop handlers first to handle an `Add` operation in the catalog. It is recommended that you convert existing `AdditionListeners` to drop handlers.

This section describes the procedure to create and register drop handlers in your application. It contains the following subsections:

- Section 21.7.1, "How to Create and Register Java Drop Handlers"
- Section 21.7.2, "What Happens at Runtime"

21.7.1 How to Create and Register Java Drop Handlers

This section describes how to create a sample drop handler for adding XML content from the Resource Catalog and register the drop handler with Composer. It includes the following subsections:

- Section 21.7.1.1, "Creating a Drop Handler"
- Section 21.7.1.2, "Registering a Drop Handler with Composer"

21.7.1.1 Creating a Drop Handler

To create a drop handler with Composer, you must first extend the abstract base class, `DropHandler`, and implement the `getName()`, `getAcceptableFlavors()`, and `handleDrop()` methods. This section describes an example to add an XML

component called `Test` to the Resource Catalog and create a drop handler named `Custom XML` for adding XML content.

To create a drop handler:

1. In JDeveloper, select the **Portal** project, and from the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.
3. In the Create Java Class dialog, specify a name for the class, for example, `TestDropHandler`.
4. In the Extends field, enter or browse to select the `Drophandler` class, `oracle.adf.view.page.editor.drophandler.DropHandler`.

   Note: If you are not able to see the `oracle.adf.view.page.editor` classpath, make sure that Composer is included in the project's libraries and classpath. To do this, right-click the project in the Application Navigator and select **Project Properties**. In the Project Properties dialog, select **Libraries and Classpath**, and add **Composer** to the list.

5. Add the required import statements and click **OK**.

   The Java class source must look like the following:

```
package test;

import java.awt.datatransfer.DataFlavor;
import java.awt.datatransfer.Transferable;

import java.io.ByteArrayInputStream;

import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import oracle.adf.rc.component.XmlComponentFactory;
import oracle.adf.view.page.editor.drophandler.DropEvent;
import oracle.adf.view.page.editor.drophandler.DropHandler;

import org.apache.myfaces.trinidad.change.AddChildDocumentChange;
import org.apache.myfaces.trinidad.change.ChangeManager;
import org.apache.myfaces.trinidad.change.DocumentChange;
import org.apache.myfaces.trinidad.context.RequestContext;

import org.w3c.dom.Document;
import org.w3c.dom.DocumentFragment;

public class TestDropHandler extends DropHandler {
    public TestDropHandler() {
        super();
    }

    public String getName() {
        return null;
    }
```

```
    public DataFlavor[] getAcceptableFlavors() {
        return new DataFlavor[0];
    }

    public boolean handleDrop(DropEvent dropEvent) {
        return false;
    }
}
```

6. Implement the `getName()` method as follows to return the name of the drop handler:

```
public class TestDrophandler extends DropHandler {
    public TestDrophandler() {
        super();
    }

    public String getName() {
        return "Custom XML";
    }
. . .
```

This value (`Custom XML`) appears in the context menu on the Add link next to the XML component in the Resource Catalog.

7. Implement the `getAcceptableFlavors()` method as follows to get a list of supported data flavors for the XML component:

```
public class TestDropHandler extends DropHandler
{
  private static final DataFlavor[] ACCEPTABLE_FLAVORS =
  { XmlComponentFactory.XML_STRING_FLAVOR };

    . . .

    public DataFlavor[] getAcceptableFlavors() {
    return ACCEPTABLE_FLAVORS;
    }
    . . .
```

8. Implement the `handleDrop(DropEvent)` method as shown in the following sample file to handle the drop event and add the XML component to the page.

```
public class TestDropHandler extends DropHandler
{
  . . .

  public boolean handleDrop(DropEvent de) {
    Transferable transferable = de.getTransferable();
    UIComponent container = de.getContainer();
    int index = de.getDropIndex();

    try {
        FacesContext context = FacesContext.getCurrentInstance();
        RequestContext rctx = RequestContext.getCurrentInstance();

        String fragMarkup = null;

        // Get the TransferData from the Transferable (expecting a String)
        Object data = getTransferData(transferable);
```

```
        if (data instanceof String) {
          fragMarkup = "<?xml version='1.0' encoding='UTF-8'?>" + (String)data;
        } else {
            return false;
        }

        // Get a DocumentBuilder
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        factory.setNamespaceAware(true);
        factory.setValidating(false);
        DocumentBuilder builder = factory.newDocumentBuilder();

        // Parse the xml string into a Document object using DocumentBuilder
        byte[] markupBytes = fragMarkup.getBytes();
        Document newDoc =
        builder.parse(new ByteArrayInputStream(markupBytes));

        // Transform the Document into a DocumentFragment
        DocumentFragment docFrag = newDoc.createDocumentFragment();
        docFrag.appendChild(newDoc.getDocumentElement());

        // Create an "add child" document change, that is, insert the fragment
        DocumentChange change = null;
        if (index < container.getChildCount()) {
        // Get the ID of the component we'll be adding just before
           String insertBeforeId = container.getChildren().get(index).getId();
           change = new AddChildDocumentChange(insertBeforeId, docFrag);
        } else {
            change = new AddChildDocumentChange(docFrag);
        }

        // Apply the "add child" DocumentChange using ChangeManager
        ChangeManager changeManager = rctx.getChangeManager();
        changeManager.addDocumentChange(context, container, change);

        // Refresh the target container using PPR
        rctx.addPartialTarget(container);

        // Mark the drop as completed.
        return true;
        } catch (Exception e) {
            return false;
        }
    }
```

In this example, a `DropEvent` parameter is passed to the `handleDrop` method. This parameter has three attributes that can be described as follows:

- `transferable`, like `DataFlavor`, is a standard Java class that contains the data being added.
- `container` is the container into which component must be dropped.
- `index` is the position of the component inside the container. For example, first, second, and so on.

9. Save the `TestDropHandler.java` file.

21.7.1.2 Registering a Drop Handler with Composer

After implementing the drop handler, you must register it with Composer. Registration is necessary for ensuring that the handler is called by Composer when a user clicks an Add link in the catalog. Register drop handlers in the Composer extension file, `/META-INF/pe_ext.xml`. For more information about creating this file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

To register a drop handler:

1. In the `pe_ext.xml` file, add the following entries:

```
<drop-handlers>
  <drop-handler>test.TestDropHandler</drop-handler>
</drop-handlers>
```

   where `TestDropHandler` is the name of the drop handler implementation.

2. Save the file.

You can register any number of drop handlers in the extension file by adding that many `<drop-handler>` elements.

21.7.1.3 Adding an XML Component to the Resource Catalog

Since you created a drop handler for XML components generated by the `XmlComponentFactory` class, you can test how the drop handler works at runtime by adding an XML component to the Resource Catalog and then adding that component to your page at runtime.

To add an XML component to the catalog:

1. Open the default catalog definition file, `default-catalog.xml`.

   For information about the location of this file, see "Default Resource Catalog Configuration".

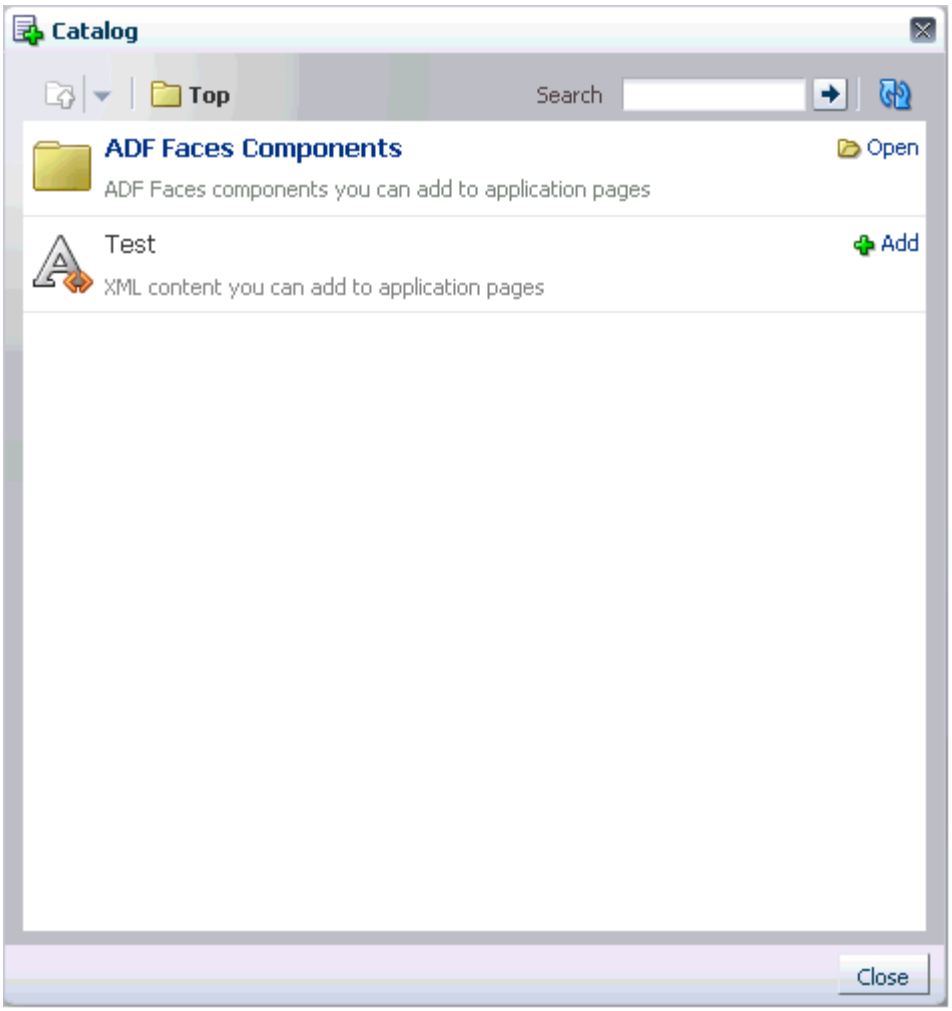
2. Add the following code within the `<contents>` section of the file:

```
<component id="pc" factoryClass="oracle.adf.rc.component.XmlComponentFactory">
  <attributes>
    <attribute attributeId="Title" value="Test"/>
    <attribute attributeId="Description" value="XML content you can add to
application pages"/>
    <attribute attributeId="IconURI" value="/adf/pe/images/elementtext_
qualifier.png"/>
    </attributes>
    <parameters>
      <parameter id="xml">
        <af:outputText value="Hello!"
xmlns:af="http://xmlns.oracle.com/adf/faces/rich"/>
      </parameter>
    </parameters>
</component>
```

3. Save the `default-catalog.xml` file.

   At runtime, the Resource Catalog shows the XML component, as shown in Figure 21–17. For more information, see Section 16.2.3.4, "Adding a Component."

Figure 21–17 XML Component in the Catalog

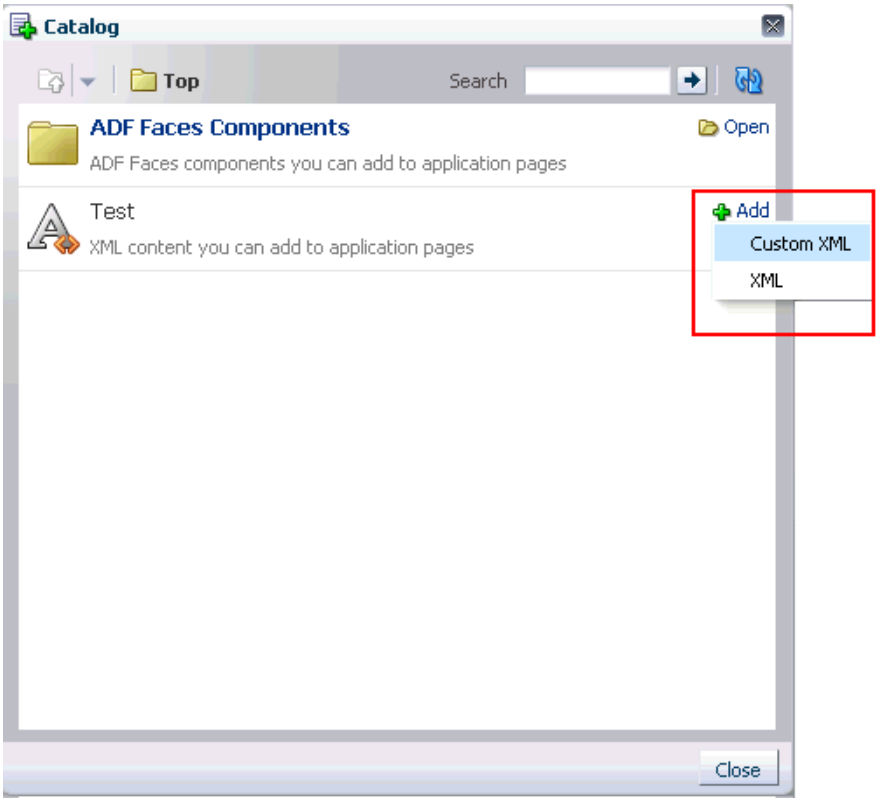


21.7.2 What Happens at Runtime

When a user clicks the Add link against a component in the catalog, all drop handlers supporting that flavor are displayed as options on the context menu of the Add link. However, if only a single drop handler is available to handle that flavor, then the resource is added to the page immediately.

Figure 21–18 shows the context menu displayed on clicking the Add link of the `Test` XML component with the `Custom XML` option and the default XML option.

Figure 21–18 Drop Handlers on an XML Component in the Catalog



21.8 Defining Property Filters

Some component properties are not displayed in Composer's Component Properties dialog because they are filtered out by default. The default filters are defined in the `<filter-config>` section of the Composer's extension file (/META-INF/pe_ext.xml).

Global filters filter attributes for all components. They are defined using the `<global-attribute-filter>` tag. Tag-level filters filter attributes for a specified component only. They are defined using the `<taglib-filter>` tag.

> **Note:** In an extension file, you can have any number of `<taglib-filter>` tags under `<filter-config>`, but you can have only one `<global-attribute-filter>` tag to define all global attribute filters.

You can define additional filters to hide more properties in the Component Properties dialog. This section describes how. It contains the following subsections:

- Section 21.8.1, "How to Define Property Filters"
- Section 21.8.2, "What Happens at Runtime"
- Section 21.8.3, "How to Remove Property Filters"

21.8.1 How to Define Property Filters

Composer defines a built-in filter configuration. You can use the extension file, `pe_ext.xml`, to define additional property filters and to delete filters. You can define any number of filters, even for a single tag, in different extension files. Composer merges the filtering information from all the extension files.

> **Note:** For information about creating the `pe_ext.xml` file, see Section 21.2.1.2, "Registering Add-Ons with Composer."

To define property filters in an extension file, use the `filter-config` element. The following elements are defined for the contained `attribute` tags.

- The `label` element allows you to override the default label for the element in the UI. You can use EL to retrieve localized strings or display different labels in design and source view.
- The `view` element can be used to filter attributes in specific tags for a specific view (`design` or `source`).
- The `filtered` element can be used to display an attribute even if it is globally filtered. To restore a globally filtered element, use `filtered="false"`.

In the `pe_ext.xml` file, add the `filter-config` element as shown in the following example:

```
<filter-config>
   <global-attribute-filter>
    <attribute name="readOnly" filtered="false" label="Read Only"/>
    <attribute name="required" filtered="false" label="Mandatory"/>
    <attribute name="shortDesc" filtered="false" label="#{Bundle.PNL_TB_
SHORTDESC}"/>
  </global-attribute-filter>

  <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
    <tag name="goLink">
      <attribute name="accessKey" view="design"/>
      <attribute name="targetFrame" filtered="false" view="source"/>
      <attribute name="destination" filtered="false"
label="#{pageEditorPanelBean.layoutView ?  'Where do you want to surf?' :
'Destination'}"/>
    </tag>
    <tag name="inputText">
      <attribute name="all" view="design"/>
      <attribute name="label" label="Field Name" filtered="false" />
      <attribute name="readOnly" filtered="false" />
      <attribute name="required" filtered="false" />
      <attribute name="rendered" filtered="false" />
      <attribute name="wrap" filtered="false" />
    </tag>
  </taglib-filter>
</filter-config>
```

Save the file.

21.8.2 What Happens at Runtime

At runtime, when you edit a component's properties, the properties that were filtered out are not rendered in the Component Properties dialog.

21.8.3 How to Remove Property Filters

You can remove global and tag-level filters so that previously filtered properties are now rendered in the Component Properties dialog. This is useful for displaying properties that are filtered out by Composer's built-in filters or by another extension file defined elsewhere the application.

> **Note:** After you remove a property filter, it is rendered in the Component Properties dialog even if a filter is defined for that property in another extension file.

To remove a property filter:

1. Edit the Composer extension file, `pe_ext.xml`, available in the `META-INF` directory.

   You can remove property filters by editing entries in this file.

2. Search for the attribute from which to remove the filter, and set `filtered` to `false` in the `<attribute>` tag as shown in the following example:

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">
. . .
<filter-config>
  <global-attribute-filter>
    <attribute name="accessKey" filtered="false" />
    <attribute name="attributeChangeListener" />
    . . .
  </global-attribute-filter>
  <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
    <tag name="activeCommandToolbarButton">
      . . .
      <attribute name="windowWidth" filtered="false"/>
    </tag>
  </taglib-filter>
</filter-config>
</pe-extension>
```

3. Save the file.

21.9 Enabling Parameter Support on the Customization Manager Task Flow

If you enable parameter passing on the task flow, administrators can customize Customization Manager by setting certain parameters. By default, an administrator cannot pass parameters to the Customization Manager task flow in Composer. Table 21–2 describes the parameters supported by the Customization Manager task flow.

Table 21–2 Customization Manager Task Flow Parameters

Parameter Name	Value	Description
defaultPage	Page URI	Used to specify the default page to be managed. If this value is null, the current page will be used.

Table 21–2 (Cont.) Customization Manager Task Flow Parameters

Parameter Name	Value	Description
dynamicNodeMap	`Map<String,List<CustomizationManagerNode>>`	Used to return a list of pages, XML documents, or task flows on a given page or fragment, using the following format: `Page1.jspx -> List of {CustomizationManagerTaskFlowNode or CustomizationManagerPageNode or CustomizationManagerXMLNode }` Use `CustomizationManagerTaskFlowNode` to pass dynamic taskflows in that page. Use `CustomizationManagerPageNode` to pass page template of the page. Use `CustomizationManagerXMLNode` to pass any XML documents owned by this page.
showBase	true/false Default: false	Used to show or hide the column for the base document.
showCurrentLayers	true/false Default: true	Used to show or hide the column showing details about the currently used layer.
showAllLayers	true/false Default: true	Used to show or hide the column showing details about layers other than the current one.
showLayersforCurrent	`Map<String,List<String>>`	Used to specify a list of layers that must be available in the Current Context column. This parameter takes the layer name and list of layer values and patterns that must be shown in the Current Context column. See Example 21–16 for more information.
showLayersforAll	`Map<String,List<String>>`	Used to specify a list of layers that must be available in the All Layers column. This parameter takes the layer name and list of layer values and patterns that must be shown in the All Layers column. See Example 21–16 for more information.
labelPrefixes	`List<String>`	Used to specify a list of space-separated label prefixes that must be displayed in the Promote dialog. Only those labels created with the specified prefixes are displayed in the Promote dialog. By default, all the labels are displayed.

Table 21–2 (Cont.) Customization Manager Task Flow Parameters

Parameter Name	Value	Description
maxDepth	String	Used to restrict the depth of a task flow hierarchy in the Customization Manager. The default value is `10`. Consequently, if you have a task flow with cyclic dependency, this parameter prevents a stack overflow by limiting the depth to ten nested task flows. You can display a maximum nesting of 10 task flows. Customization Manager displays only a depth of 10 even if you specify avalue higher than 10.
showDownload	true/false Default: true	Used to show or hide the Download link.
showUpload	true/false Default: true	Used to show or hide the Upload link.
showDelete	true/false Default: true	Used to show or hide the Delete link.
showPromote	true/false Default: true	Used to show or hide the Promote link.

Note: The implementation of `Map` and `List` must be `Serializable` for Customization Manager to work correctly in a cluster-based environment. Typically, `ArrayList` and `HashMap` are `Serializable`.

This section describes the steps to enable parameter editing on the Customization Manager task flow. It contains the following subsections:

- Section 21.9.1, "How to Set the Customization Manager Task Flow Parameters"
- Section 21.9.2, "What Happens When You Enable Parameter Support"

21.9.1 How to Set the Customization Manager Task Flow Parameters

To pass parameters to the Customization Manager task flow, you must define the `parameters` attribute as part of the Customization Manager add-on panel configuration in your application's `adf-config.xml` file.

To enable parameter editing on the task flow:

1. Open the application's `adf-config.xml` file, located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the following namespace within the `adf-config` element in the file:

   ```
   xmlns:pe="http://xmlns.oracle.com/adf/pageeditor/config"
   ```

3. Add a `<pe:page-editor-config>` entry with the namespace and include an `<pe:addon-panels>` entry.

Within <pe:addon-panels>, add an <pe:addon-panel> entry for the default Customization Manager panel and define the parameters attribute by setting it to appUtilBean.customizationManagerParams, as shown in Example 21–16.

Example 21–16 Customization Manager Add-On Referenced in adf-config.xml

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:addon-panels>
    <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.customization-manager"
parameters="#{AppUtilBean.customizationManagerParams}"/>
    . . .
  </pe:addon-panels>
</pe:page-editor-config>
```

where AppUtilBean is a sample implementation of the AppUtilBean class, and is as shown in the following example:

```
public Map<String, Object> getCustomizationManagerParams()
{
  Map<String, Object> paramMap = new HashMap<String, Object>();
  Map<String, List<String>> showLayersForCurrent = new HashMap<String,
List<String>>();
  showLayersForCurrent.put("user", Arrays.asList("admin%", "weblogic"));
  paramMap.put("showLayersForCurrent", showLayersForCurrent);

  Map<String, List<String>> showLayersForAll = new HashMap<String,
List<String>>();
  showLayersForAll.put("enterprise", Arrays.asList("Frys"));
  paramMap.put("showLayersForAll", showLayersForAll);

  return paramMap;
}
```

4. Save the adf-config.xml file.

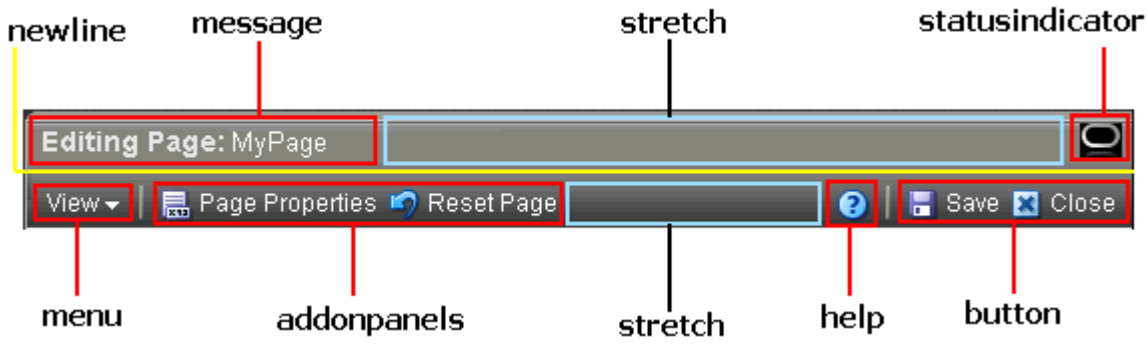
21.9.2 What Happens When You Enable Parameter Support

Customization Manager displays options to users based on the new parameter settings.

21.10 Customizing the Composer Toolbar

You can customize the default Composer toolbar by adding, removing, or rearranging elements and overriding existing elements. To enable toolbar extensibility, the Composer toolbar elements have been grouped into sections. You can work with these sections to customize your toolbar. Each section has a specific name and contains one or more elements inside it, as shown in Figure 21–19.

Figure 21–19 Composer Toolbar



Notes: The Save button is displayed on the toolbar only if the application is configured to use a sandbox. For more information, see Section 23.2, "Using Composer Sandbox."

The Help icon is displayed on the toolbar only if you have configured a help provider in your application or hooked up help from Composer panels or dialogs. For more information, see Section B.4, "Composer Help Topic IDs."

The Composer toolbar is created using the ADF Faces `Toolbox` tag. Each row in the toolbar is a `Toolbar` tag. Consequently, UI components such as separator bars are inherited from the `Toolbar` component.

Table 21–3 describes the toolbar areas.

Table 21–3 Toolbar Areas

Section or Area	Description
`message`	Area containing the message with the page name.
`statusindicator`	Area containing the status indicator.
`menu`	Area containing the View menu that allows users to switch between Design view and Source view.
`addonpanels`	Area containing Composer add-ons. The add-ons are all displayed in a sequence within this area. For more information, see Section 21.2, "Creating Composer Add-Ons."
`help`	Area containing the Help icon.
`button`	Area containing the Save and Close buttons.
`stretch`	Wide space between two sections.
`newline`	A line break between sections. All sections after a line break are pushed to a new line on the toolbar.

Using toolbar sections, you can customize the Composer toolbar in the following ways:

- Rearrange and hide built-in elements
- Add new sections with custom elements

- Modify default sections to display custom elements

This section describes the steps to customize the Composer toolbar. It contains the following sections:

- Section 21.10.1, "How to Rearrange or Hide Toolbar Elements"
- Section 21.10.2, "What Happens at Runtime"
- Section 21.10.3, "How to Add New Sections to the Toolbar"
- Section 21.10.4, "How to Override a Toolbar Section to Display Custom Content"

21.10.1 How to Rearrange or Hide Toolbar Elements

Use the `toolbarLayout` attribute on a `Page Customizable` component to control which toolbar sections are displayed and the order in which they appear.

To customize the toolbar for all editable pages in the application, you can create a template, add a `Page Customizable` component to the template, and specify the `toolbarLayout` attribute against the `Page Customizable` component. You can then base all the pages in the application on that template.

If you do not specify a value for `toolbarLayout`, this attribute is internally set to `message stretch statusindicator newline menu addonpanels stretch help button`, which is the default layout for the Composer toolbar sections.

To customize the toolbar on an application page:

1. Open your customizable JSPX page in JDeveloper and select the `Page Customizable` component in the Structure window.
2. In the Property Inspector, specify space-separated values for the `toolbarLayout` attribute.

   The section names in Table 21–3 are valid values for this attribute.

   Note: You can add only one `stretch` value per row on the toolbar. If you add more than one `stretch` value, only the first one is displayed; all others are ignored.

   In Source view, the `toolbarLayout` attribute appears as shown in the following example:

```
<pe:pageCustomizable id="pageCustomizable1"
                     toolbarLayout="button stretch statusindicator menu"
  <cust:panelCustomizable id="panelCustomizable1" layout="scroll"/>
  <f:facet name="editor">
    <pe:pageEditorPanel id="pep1"/>
  </f:facet>
</pe:pageCustomizable>
```

3. Save the JSPX file.

21.10.2 What Happens at Runtime

At runtime, the toolbar displays the sections you specified in the order you specified them (Figure 21–20).

Figure 21–20 Customized Composer Toolbar



21.10.3 How to Add New Sections to the Toolbar

You can add custom sections by creating facets and specifying the facet names in the `toolbarLayout` attribute. Populate the new facets with elements you want to display on the toolbar. The following example shows you how to create a section; specifically, how to add a **Report a Bug** button that opens a popup that enables users to report a bug.

To add a custom section:

1. Open the JPSX page for which you want to customize the toolbar.
2. Add a `facet` inside the `Page Customizable` and name it `bugreport`.
3. Add a `Command Toolbar Button` component inside the facet and set the `text` attribute to `Report a Bug`.
4. To add an image to the button, specify the path to the image using the `icon` attribute.
5. Add the `Show Popup Behavior` component to invoke a popup on clicking the `Command Toolbar Button`.

   In Source view, the `Page Customizable` would appear as follows:

```
<pe:pageCustomizable id="pageCustomizable1"
             toolbarLayout="message stretch bugrep newline menu stretch button"
  <f:facet name="bugrep">
    <af:commandToolbarButton id="cmd1" text="Report a bug"
                             icon="/images/bug.png"
                             clientComponent="true">
      <af:showPopupBehavior popupId="p1" triggerType="action"/>
    </af:commandToolbarButton>
  </f:facet>
. . .
  <af:popup id="p1">
    <af:dialog id="dlg1" title="File a Bug"
               affirmativeTextAndAccessKey="File">
      <af:panelFormLayout labelWidth="30%" id="pfl1">
        <af:inputText id="it0" maximumLength="80"
                      label="Product/Component" required="true"/>
        <af:inputText id="it1" maximumLength="80" label="Subject"
                      required="true"/>
        <af:inputText id="it2" rows="3" label="Problem" required="true"/>
        <af:inputText id="it3" rows="3" label="Steps" required="true"/>
      </af:panelFormLayout>
    </af:dialog>
  </af:popup>
  . . .
</pe:pageCustomizable>
```

6. Save the page.

   At runtime, the Composer toolbar displays a **Report a Bug** button, as shown in Figure 21–21.

Figure 21–21 Composer Toolbar with Custom Section



Clicking the Report a Bug button displays the File a Bug dialog that enables users to report a bug.

21.10.4 How to Override a Toolbar Section to Display Custom Content

You can display custom content in a default toolbar section by adding a facet of the same name as the section and populating it with custom content. The facet content overrides the content of the default section. For example, to display a custom message to users in place of the `Editing Page: Page_Name` message, you can create a custom facet named `message` and include an `Output Text` component with the text you want to display to users. The `Output Text` component then displays in place of the default message. The following example shows a `Page Customizable` component with a custom toolbar message:

```
<pe:pageCustomizable id="pageCustomizable1"
                     toolbarLayout="message newline button stretch statusindicator
menu"
  <cust:panelCustomizable id="panelCustomizable1" layout="scroll"/>
  <f:facet name="message">
    <af:outputText value="Welcome!"/>
  </f:facet>
  <f:facet name="editor">
    <pe:pageEditorPanel id="pep1"/>
  </f:facet>
</pe:pageCustomizable>
```

Figure 21–22 shows the Composer toolbar displayed for the page at runtime.

Figure 21–22 Composer Toolbar with a Custom Message



21.11 Enabling Direct Select in Design View

Direct select allows you to extend specific customization capabilities from Composer Source view to Design view, allowing users to modify elements from within the page at runtime. This allows you to put more control in the hands of the business user, while creating a safe environment where no damage can be done to the underlying application.

All direct select functionality is available in Source view, but business users do not have the knowledge or experience to use Source view safely and effectively. These users could easily break the page and circumvent application logic.

Using direct select, you can enable a small number of chosen components to be selectable and editable. For example, changing the order of fields in a form, changing the labels on form fields, changing column names in a table, or hiding specific fields in a form.

For more information on direct select functionality, see Section 21.11.2, "What Happens at Runtime."

21.11.1 How to Enable and Configure Direct Select

This section explains how to enable and configure direct select in your application.

1. First you must enable direct select for the Design view in Composer. In adf-config.xml, inside the `<pe:page-editor-config>` tag, add the `<pe:enable-design-views>` tag to define the available views. The possible constant values are:

   - `all`: Makes all views available and displays a toolbar with buttons to switch between views.
   - `add-content` (default): Makes the legacy Design view the only available view.
   - `select`: Makes the Select view the only available view.
   - `select add-content`: Limits the available views to only the Select and legacy Design views (additional views will be added in future releases). Displays the toolbar with buttons to switch between views.

   You can also use EL to determine the appropriate view, as shown in the example below:

   ```
   <pe:enable-design-views>#{testBean.designViews}</pe:enable-design-views>
   ```

   Note: This setting defines the default for the application. You can override this setting in any `pageCustomizable` tag using the `designViews=""` attribute.

   By default the hot key to open and close Design view is Ctrl-Shift-S. To change or disable the hot key, add a `<pe:component-view-switch-key>` tag inside the `<pe:page-editor-config>` tag. For example:

   ```
   <pe:component-view-switch-key>
     <pe:enabled>true</pe:enabled>
     <pe:key-sequence>ctrl shift l</pe:key-sequence>
   </pe:component-view-switch-key>
   ```

2. In pe-ext.xml, add the `<selection-config>` tag to enable or disable direct select for specific components and define associated operations. To make a component selectable, you must define operations within this tag. (If no operations are defined and selection is enabled for a component, selecting the component will display an empty frame.)

   Within the `<selection-config>` tag, you can configure direct select globally using `<global-filter>` and by namespace and tag using `<selection-taglib-filter>`. The `<selection>` tag has two required attributes:

   - `view`: Defines the view to be configured in the tag. In the current release, the only accepted value is `design` (direct select cannot be disabled in source view).
   - `enabled`: Sets whether or not direct select is enabled in the specified view.

   Available operations are defined within `<operation>` tags under each specific component. The following elements are used to define operations:

- `name`: Defines the operation (standard or custom) to be executed. The operation name is the same as the property panel ID. For a list of standard operations, see Section B.3.2, "Default Property Panels." For custom operations, the name is the same as the custom panel ID. For details on custom operations, see Section 21.3, "Creating Custom Property Panels."
- `label` (optional): Sets the tab label in the popup window. The default label is the operation name.
- `filtered`: Set to `false` to include the operation in the popup window.

Note: It is a recommended practice to provide selection for a small number of components at first, then add more as functionality is tested and learned. This can be done easily by globally disabling selection in the `<global-filter>` tag and enabling selection for individual components using `<selection-taglib-filter>` as shown in the example that follows.

```
<selection-config>
  <global-filter>
    <selection view="design" enabled="false"/>
  </global-filter>

  <selection-taglib-filter
   namespace="http://xmlns.oracle.com/adf/faces/customizable">
    <tag name="showDetailFrame">
      <selection view="design" enabled="true"/>
      <operation name="oracle.adf.pageeditor.pane.sdfprop"
                 label="SDF Change Property"
                 filtered="false"/>
    </tag>
  </selection-taglib-filter>

  <selection-taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
    <tag name="goLink">
      <selection view="design" enabled="true"/>
      <operation name="oracle.adf.pageeditor.pane.generic-property-inspector"
                 label="Change Property"
                 filtered="false"/>
      <operation name="oracle.adf.pageeditor.pane.inline-style-editor"
                 label="Inline Style"
                 filtered="false"/>
    </tag>
    <tag name="commandButton">
      <selection view="design" enabled="true"/>
      <operation name="oracle.adf.pageeditor.pane.generic-property-inspector"
                 label="Change Property"
                 filtered="false"/>
    </tag>
  </selection-taglib-filter>
</selection-config>
```

To further simplify the user experience, direct select allows you to filter specific attributes for a tag so that a smaller set is displayed. For details, see Section 21.8.1, "How to Define Property Filters."

3. To enable or disable direct select for a specific component within a .jspx page, add the `<f:attribute name="pe-select" value="true|false">` tag within the component tag as shown in the example that follows. The setting in this tag

overrides the setting in the `<selection-config>` for the component in the pe_ext.xml file. (As noted above, to enable direct select, operations must be defined for the component in the pe_ext.xml file, or selecting it will open an empty dialog.)

```
<af:panelGroupLayout id="pgl2" layout="vertical">
<f:attribute name="pe-select" value="false"/>
  <af:inputText id="it6" label="Unselectable" value="Enter here..."/>
  <af:goLink id="gl1" destination="http://www.oracle.com" text="Oracle"/>
  <af:inputText id="it7" label="Selectable(override)" value="Enter here...">
    <f:attribute name="pe-select" value="true"/>
  </af:inputText>
</af:panelGroupLayout>
```

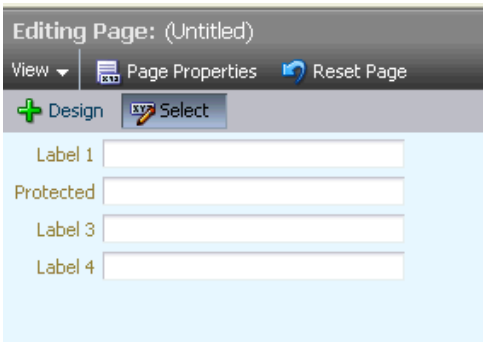
4. To disallow the use of Expression Language in form fields and protect existing EL from overwrite, see Section 21.4.2, "How to Protect Expression Language."

For more information on specific elements, see the following sections in Appendix B, "Composer Component Properties and Files":

- Section B.2.1.8, "selection-config"
- Section B.2.2.5, "enable-design-views"
- Section B.2.2.6, "allow-el and protect-el"

21.11.2 What Happens at Runtime

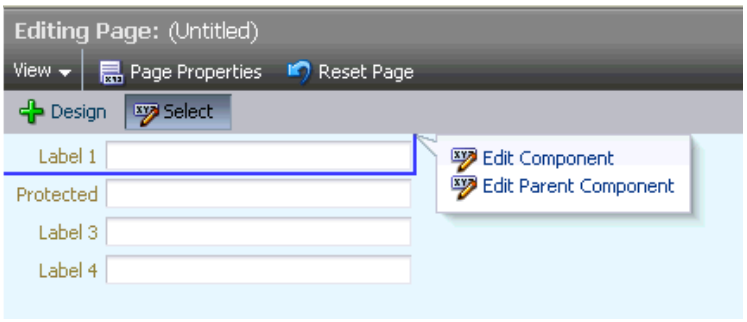
When direct select is enabled, the outward presentation of the page at runtime does not appear any different to the end user. When a user with the appropriate permissions clicks Ctrl-Shift-S, the page opens in Design view.



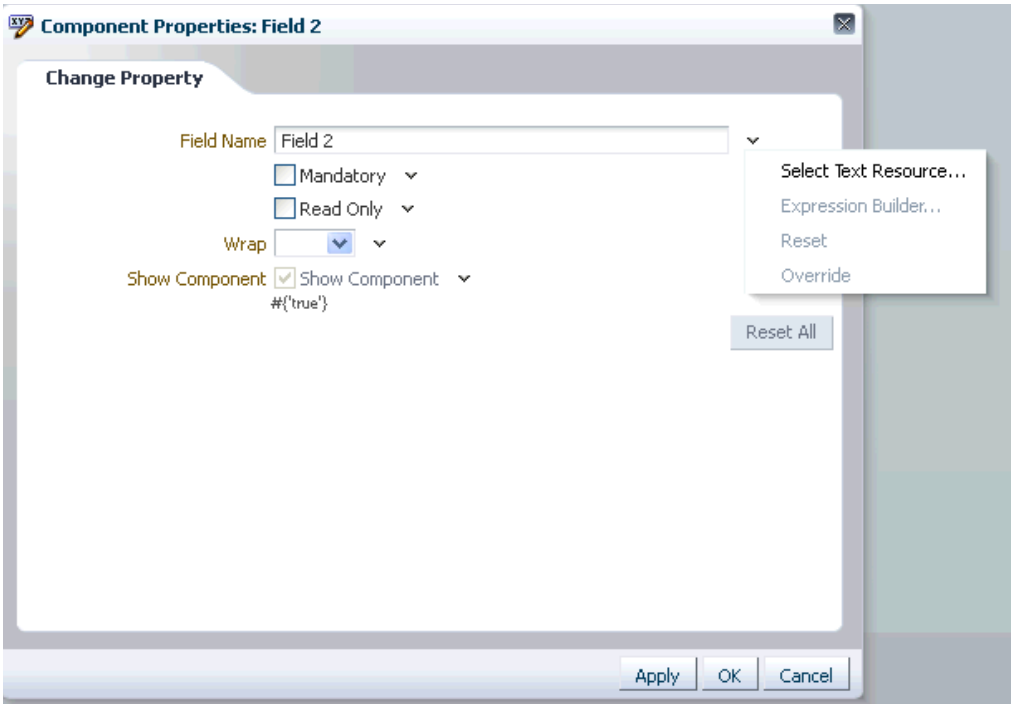
Design view includes the Composer toolbar. If more than one view is enabled, the page also includes a second toolbar with **Design** and **Select** buttons to switch between views, as shown in the image above. If only a single view is enabled, these buttons are not displayed.

> **Note:** If 'select' is defined as the only available view, users have no way to add content from Design view. If 'add-content' is defined as the only available view, the user experience is the same as if direct select was not enabled.

To edit the properties for a component, the user selects an enabled element and clicks **Edit Component** or **Edit Parent Component** (when applicable).



The **Component Properties** popup window allows the user to edit properties defined for the component and access other operations configured for the component. If the user chose Edit Parent Component, a Manage Children tab is included in the window. If EL is not allowed or is protected from overwrite, the Expression Builder and Override options are not available, as shown in the image below. This image also includes an example of a field that is protected from edit (Show Component).



Multiple operations are shown in tabs in the popup window. The order of tabs is defined in the panel registration. If an operation references a panel that is not registered for the enclosing component, the tab for that operation will not be displayed. For details on panel registration, see Section 21.3, "Creating Custom Property Panels."

21.12 Troubleshooting Problems with Composer Extensibility Features

This section provides information to assist you in troubleshooting problems you may encounter while using the Composer extensibility features.

For information about configuring logging, see "Configuring ADF Logging for Composer".

Problem

You created an add-on, but it does not appear on the Composer toolbar.

Solution

Ensure the following:

- The `pe_ext.xml` file is in `/META-INF` folder in a JAR file or the application and is available in the class path.
- The task flow binding ID specified while registering the panel in `pe_ext.xml` is correct.
- An `<pe:addon-panel>` entry exists under the `<pe:page-editor-config>` section in the `adf-config.xml` file.
- If the add-on is in a JAR file, ensure that the JAR file is created as an ADF Library.

For information about add-ons, see Section 21.2, "Creating Composer Add-Ons."

Problem

You have registered a custom property panel. However, it does not appear in the Component Properties dialog when your select a component to display its properties.

Solution

Ensure the following:

- The `pe_ext.xml` file is in `/META-INF` folder in a JAR file or the application and is available in the class path.
- The task flow binding ID specified while registering the panel in `pe_ext.xml` is correct.
- The `property-panel` registration is correct and is specified against the component you want the panel to appear against. Further, the fully qualified class name of the component is correctly specified using the `component` node.
- A duplicate property panel registration is not overriding your panel entry.
- If the panel is configured to use the rendered attribute, ensure that the value or EL evaluates to `true`.
- If the add-on is in a JAR file, ensure that the JAR file is created as an ADF Library.

For information about custom property panels, see Section 21.3, "Creating Custom Property Panels."

Problem

You do not see some properties of a component in the Component Properties dialog.

Solution

Ensure that the component properties are not filtered or restricted. For more information, see Section 21.8.1, "How to Define Property Filters" and Section 24.4, "Applying Attribute-Level Security."

22

Performing Advanced Composer Configurations

This chapter describes how to perform certain advanced Composer configurations to enhance the end user experience. It contains the following sections:

- Section 22.1, "Enabling Custom Actions on a Show Detail Frame Component by Using Facets"
- Section 22.2, "Enabling Custom Actions on a Show Detail Frame Component By Using Facets: Example"
- Section 22.3, "Enabling Custom Actions on a Task Flow"
- Section 22.4, "Enabling Custom Actions that Display on Task Flows in the Component Navigator"
- Section 22.5, "Configuring a Keyboard Shortcut to Access Composer"
- Section 22.6, "Creating Event-Enabled Task Flows"
- Section 22.7, "Configuring an Application Page to Display in Source View by Default"
- Section 22.8, "Disabling the Source View Menu for an Application"
- Section 22.9, "Disabling Task Flow Zoom Capability"
- Section 22.10, "Applying Styles to Components"
- Section 22.11, "Configuring the Persistence Change Manager"
- Section 22.12, "Configuring Runtime Resource String Editing"

22.1 Enabling Custom Actions on a Show Detail Frame Component by Using Facets

You can use the `Show Detail Frame` component's facets to define and display custom actions on `Show Detail Frame` components. For example, if your `Show Detail Frame` contains a list of services provided in your application, you can add a custom action, `Show Detailed Information`, which opens up a task flow containing details about the various services.

For information about the facets supported by `Show Detail Frame` components, see Section B.1.5, "Show Detail Frame Component."

Oracle JDeveloper displays all facets available to the `Show Detail Frame` component in the Structure window, however, only those that contain UI components appear activated.

To add a Show Detail Frame facet, perform the following steps:

1. Right-click a `Show Detail Frame` component in the Structure window, and select **Facets - Show Detail Frame**.
2. Click the arrow to the right of this option.
3. From the list of supported facets, select the facet you want to add.

   Note: A check mark next to a facet name means the facet is already inserted in the `Show Detail Frame` component. Selecting that facet name again would result in the facet getting deleted from the page.

   The `f:facet` element for that facet is inserted in the page.
4. Add components in the facet according to your design requirements.

   For an end-to-end example of creating and using `Show Detail Frame` facets, see Section 22.2, "Enabling Custom Actions on a Show Detail Frame Component By Using Facets: Example."

22.2 Enabling Custom Actions on a Show Detail Frame Component By Using Facets: Example

Assume a JSPX page, `Page1`, with a `Panel Customizable` component. Inside the `Panel Customizable` is a `Show Detail Frame` component (`showDetailFrame1`). Inside the `Show Detail Frame` is an ADF task flow. The `Panel Customizable` has two other `Show Detail Frame` components, one above and the other below `showDetailFrame1`. The task flow displays two `Output Text` components on the page.

You can configure an `Additional Actions` facet on the `Show Detail Frame` component to display a **Customize** action on the Actions menu along with the **Move Up** and **Move Down** actions. At runtime, the **Customize** action enables users to customize the text in the `Output Text` components. This section describes the steps you take to achieve this effect. It contains the following subsections:

- Section 22.2.1, "How to Create an ADF Task Flow"
- Section 22.2.2, "How to Include an Additional Actions Facet"
- Section 22.2.3, "How to Create a Redirection Page"
- Section 22.2.4, "How to Create Navigation Rules Between Pages"
- Section 22.2.5, "What Happens at Runtime"

22.2.1 How to Create an ADF Task Flow

To create an ADF task flow:

1. From the **File** menu, select **New**.
2. In the **New** dialog, select **JSF** under Web Tier node, and select **ADF Task Flow** under Items.
3. Click **OK**.
4. In the Create Task Flow dialog, click **OK** to create a task flow definition file by accepting the default values.

5. From the **ADF Task Flow** tag library in the Component Palette, drop two view elements, **view1** and **view2**, onto the task flow definition file.
6. Drop a **Control Flow Case** from `view1` to `view2`.
7. Click the first view element and then click the second view element to draw the control flow line.

   Name this control flow case `next`.
8. Similarly, drop a **Control Flow Case** from `view2` back to `view1` and name it `prev`.
9. Create a backing bean called `BackingBean.java` to contain values for two variables `view1` and `view2`.

   `view1` and `view2` are initialized with `initialValue1` and `initialValue2` respectively. Ensure that the code of the bean is as show in the following example:

```
package view;

public class BackingBean {
    public BackingBean() {
    }

    private String view2 ="initial Value1";
    private String view1 = "initial Value2";

    public void setView2(String view2) {
        this.view2 = view2;
    }

    public String getView2() {
        return view2;
    }

    public void setView1(String view1) {
        this.view1 = view1;
    }

    public String getView1() {
        return view1;
    }
}
```

10. In the task flow definition file, double-click **view1** to create the page fragment (`view1.jsff`) for that element.
11. Add a **Panel Group Layout** and two **Output Text** components to `view1.jsff`.
12. Specify the `Value` for the first `Output Text` component to be `#{backingBean.view1}`, and specify the `Value` for the second `Output Text` component to be `#{backingBean.view2}`.
13. Save `view1.jsff`, and close it.
14. In the task flow definition file, double-click **view2** to create the page fragment (`view2.jsff`) for that element.
15. Add just one **Output Text** component to `view2.jsff`, and specify `Value` to be `#{backingBean.view2}`.
16. Save `view2.jsff`, and close it.

22.2.2 How to Include an Additional Actions Facet

To include an Additional Actions facet:

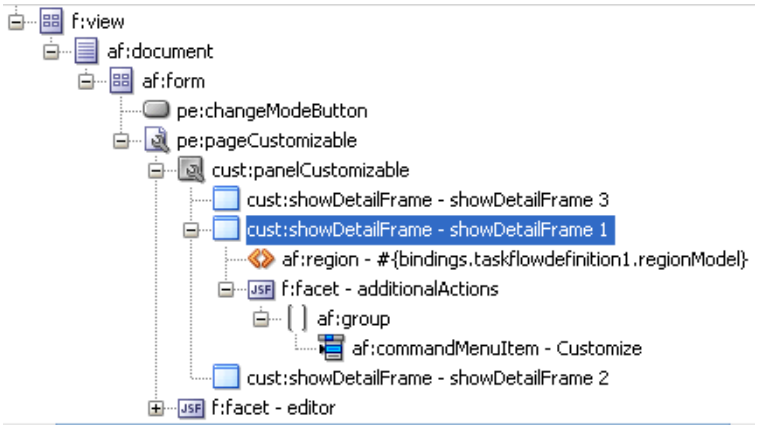
1. Create a JSPX page, `Page1.jspx`, with a `Panel Customizable` component and a nested `Show Detail Frame` component.
2. Add two more `Show Detail Frame` components, above and below the existing `Show Detail Frame` component.

   The purpose of adding three `Show Detail Frame` components is to enable the display of `Move Up` and `Move Down` actions along with the additional action on the first `Show Detail Frame` component, `showDetailFrame1`.
3. Add the task flow definition file that you created in the previous step inside `showDetailFrame1`.
4. Right-click the first `Show Detail Frame` in the Structure window, and select **Facets - Show Detail Frame**.
5. Click the arrow to the right of this option, and from the list of supported facets, select **Additional Actions**.

   The `f:facet-additionalActions` element for that facet is inserted in the page.
6. Add a **Panel Group Layout** inside the `Additional Actions` facet, and add a **Button** component inside the `Panel Group Layout` component.
7. Set the `Text` attribute for the `Button` to `Customize`, and specify `customize` as the `Action` value.

   The page in the structure navigator should appear as shown in Figure 22–1.

Figure 22–1 Page1.jspx in Structure Navigator



8. Save the page.

22.2.3 How to Create a Redirection Page

To create the redirection page:

1. Create a JSPX page called `Page2.jspx`, and add two **Input Text** components and a **Button** component.

2. Specify the `Value` for the first `Input Text` component to `#{backingBean.view2}`, and set the `Value` attribute for the second `Input Text` component to `#{backingBean.view1}`.

   The purpose of adding `Input Text` components with references to the backing bean is to enable the passing of a user's input to the bean so that it can be reflected in the `Output Text` components on `Page1.jspx`.

3. On the `Button` component, set the `Text` attribute to `OK` and specify `back` as the `Action` value.

4. Save the page.

   You can now enable switching between the two pages by defining navigation rules.

22.2.4 How to Create Navigation Rules Between Pages

To define navigation rules between the pages:

1. In the Application Navigator, under the project's WEB-INF folder, double-click the **faces-config.xml** file to open it.
2. Click the **Overview** tab to view the file in Overview mode.
3. Click the **Add** button in the Managed Bean section.
4. In the Create Managed Bean dialog, specify `backingBean` as the name.
5. For the Class Name field, click the **Browse** button adjacent to it and browse to the `BackingBean` Java class that you created earlier. Click **OK**.
6. Click **OK**.
7. From the Scope list, select **session** and click **OK**.
8. Select the **Navigation Rules** tab on the page.
9. In the From Views table, select **Page1.jspx**.
10. Under Navigation Cases, select **Page2.jspx** in the To View ID column, **customize** in the From Outcome column, and **true** in the Redirect column.
11. In the From Views table, select **Page2.jspx**.
12. Under Navigation Cases, select **Page1.jspx** in the To View ID column, **back** in the From Outcome column, and **true** in the Redirect column.

    In Source view, these entries appear as follows:

```
<managed-bean>
    <managed-bean-name>backingBean</managed-bean-name>
    <managed-bean-class>view.BackingBean</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <navigation-rule>
  <from-view-id>/Page1.jspx</from-view-id>
  <navigation-case>
    <from-outcome>customize</from-outcome>
    <to-view-id>/Page2.jspx</to-view-id>
      <redirect/>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <from-view-id>/Page2.jspx</from-view-id>
  <navigation-case>
```

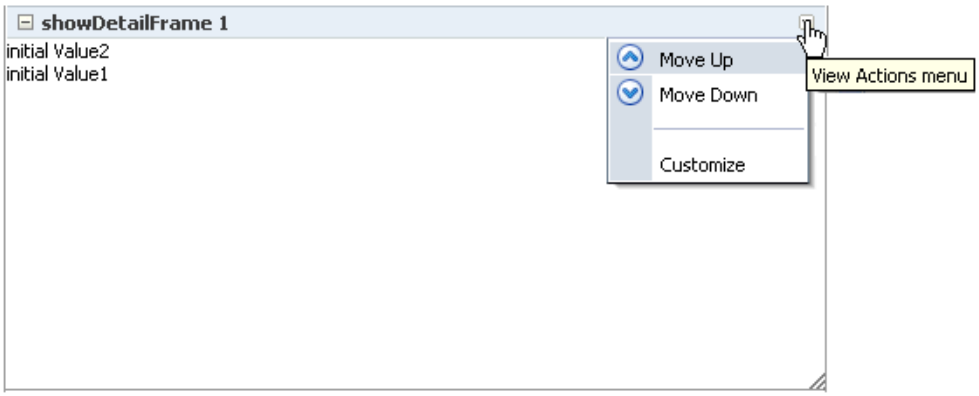
```
    <from-outcome>back</from-outcome>
    <to-view-id>/Page1.jspx</to-view-id>
      <redirect/>
    </navigation-case>
  </navigation-rule>
```

13. Save the file.

22.2.5 What Happens at Runtime

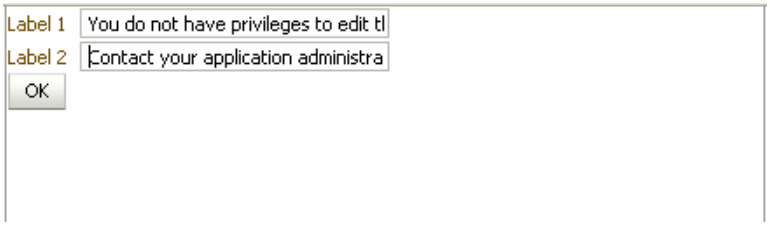
In JDeveloper, run `Page1.jspx`. The Actions menu on the `showDetailFrame1` component displays the Customize action, as shown in Figure 22–2.

Figure 22–2 Customize Action on the Actions Menu



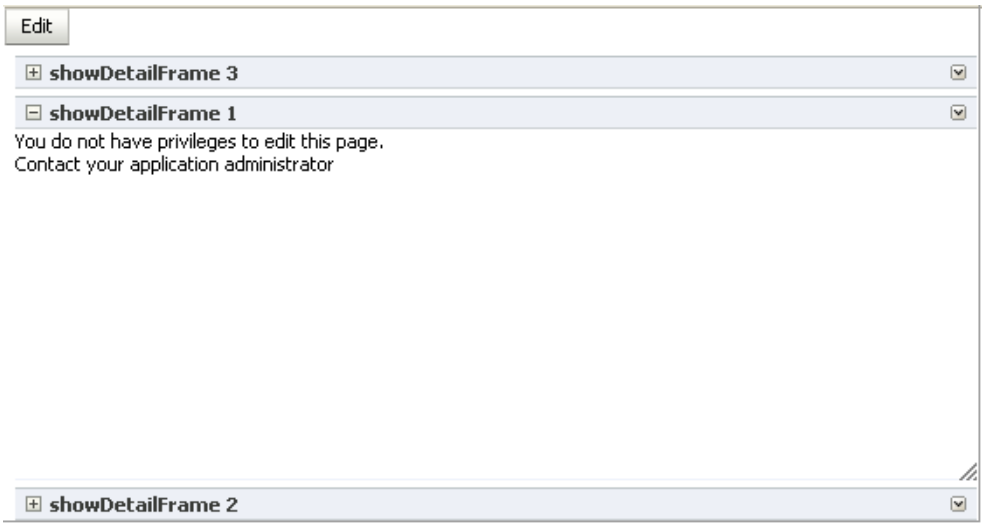
Clicking Customize takes you to `Page2.jspx` (Figure 22–4), where you can update the values for `Label1` and `Label2`.

Figure 22–3 Page with Option to Edit Text



Clicking **OK** takes you back to `Page1.jspx`, which reflects the recent text changes, as shown in Figure 22–4.

Figure 22–4 Page1 with Updated text



22.3 Enabling Custom Actions on a Task Flow

You can customize a task flow by including it in a `Show Detail Frame` component. The `Show Detail Frame` component provides certain default actions to rearrange, show, and hide components. Further, you can define custom actions to trigger the desired navigational flow within the task flow at runtime. There are two ways in which you can enable custom actions on a task flow:

- Enable custom actions directly on the task flow so that they are displayed on the `Show Detail Frame` component's Actions menu.
- Enable custom actions on the `Show Detail Frame` component enclosing the task flow so that they are displayed on the `Show Detail Frame` component's Actions menu.

This section describes both approaches. It includes the following subsections:

- Section 22.3.1, "How to Enable Custom Actions Directly on a Task Flow"
- Section 22.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow"
- Section 22.3.3, "What Happens at Runtime"
- Section 22.3.4, "How to Enable Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example"

22.3.1 How to Enable Custom Actions Directly on a Task Flow

Perform the steps in this section if you want the task flow to be self-contained, without the need to inherit global custom actions defined at the application level. You can define custom actions on a task flow by configuring a `<customComps-config>` section with a nested `<customActions>` element in the `adf-settings.xml` file. The additional custom actions specified in the `adf-settings.xml` file are displayed on the `Show Detail Frame` component that surrounds this task flow.

Typically, task flows are packaged and deployed as ADF libraries. When you create an `adf-settings.xml` file containing custom actions for a task flow, this file is also packaged in the ADF library.

To enable custom actions on a task flow:

1. If it does not already exist, create the `adf-settings.xml` file in the `META-INF` directory under the project's Web context root (for example, in the *`APPLICATION_ROOT`*`/Portal/src/META-INF` directory):

   a. From the **File** menu, select **New**.

   b. In the New Gallery dialog, expand **General**, select **XML**, then **XML Document**.

   c. Click **OK**.

   Name the file `adf-settings.xml`.

2. Add a `<custComps-config>` section in the file, with a nested `<customActions>` element.

3. Add one `<customAction>` element for each internal task flow action that you want to display as a custom action on the `Show Detail Frame` Actions menu.

   You can add any number of custom actions under the `<customActions>` element.

   Example 22–1 shows the code of the `adf-settings.xml` file with a `<customAction>` entry.

Example 22–1 The custComps-config Section in an adf-settings.xml File

```
<cust:custComps-config
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <customActions>
   <customAction action="next" location="chrome"
                 rendered="true"
                 icon="/adf/webcenter/editheader_ena.png"
                 text="Next"

taskFlowId="/WEB-INF/task-flow-definition.xml#task-flow-definition"
                 shortDesc="next"/>
    <customAction action="prev" location="chrome"
                 rendered="true"
                 icon="/adf/webcenter/editheader_ena.png"
                 text="Previous"
 taskFlowId="/WEB-INF/task-flow-definition.xml#task-flow-definition"
                 shortDesc="prev"/>
  </customActions>
 </cust:custComps-config>
```

Custom action definitions are similar at the task flow-level and application-level, except for the `taskFlowId` attribute that is used in the task flow-level setting. This attribute is used to identify the task flow on which the custom action must be defined. As an ADF library may have multiple task flows, this attribute helps identify the task flow that must render the custom action.

Notes:

- If you are defining an icon for the custom action, you must ensure that the image you specify is available in the project root folder.
- You can define custom actions for the internal actions defined in all task flows on your page; however, at runtime, the `Show Detail Frame` displays only those custom actions that correspond to the ADFc outcomes of the current view of the task flow.

4. Save the `adf-settings.xml` file.

22.3.2 How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow

You can define custom actions for a task flow on the enclosing `Show Detail Frame` component. When these actions are invoked at runtime, they trigger the desired navigational flow in the task flow. For example, you can define a custom action on a `Show Detail Frame` that specifies that the target task flow fragment opens in a separate browser window rather than inside the `Show Detail Frame`.

You can specify custom actions on `Show Detail Frame` components by adding `Custom Action` components as children of a `Show Detail Frame` component on the page. Custom actions defined in this way would be available only on the `Show Detail Frame` instance, which has the custom action as its child. Alternatively, you can specify custom actions in the application's `adf-config.xml file`. Custom actions defined in this way are available on all `Show Detail Frame` instances in the application.

This section provides information about defining custom actions on a `Show Detail Frame`. It includes the following subsections:

- Section 22.3.2.1, "Defining Custom Actions at the Instance Level"
- Section 22.3.2.2, "Defining Custom Actions at the Global Level"
- Section 22.3.2.3, "Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window"

22.3.2.1 Defining Custom Actions at the Instance Level

Define custom actions on a particular `Show Detail Frame` component instance using the `Custom Action` component. You can find the `Custom Action` component in the Composer tag library. Custom actions are stored in the JSPX page definition file of the page that contains the `Show Detail Frame`.

To define custom actions at the instance level:

1. From the Component Palette, select **Composer**.
2. Drag a **Custom Action** and drop it on the page inside the `Show Detail Frame` component, below the `af:region` element.

   Add one `Custom Action` component for each internal task flow action you want to display as a custom action on the `Show Detail Frame` Actions menu.

Note: Add `Custom Action` components below the `af:region` element. You may face problems if the region is not the first child component of the `Show Detail Frame`.

3. Define attributes for the `Custom Action` by referring to Table B–9 in Section B.1, "Composer Component Properties."

   Ensure that you populate the `Action` attribute of each `Custom Action` with the correct ADFc outcomes of the associated task flow. The code should be similar to the one shown in Example 22–2.

Example 22–2 Custom Action Code

```
<cust:showDetailFrame text="showDetailFrame 1" id="sdf1">
  <af:region value="#{bindings.taskflowdefinition1.regionModel}"
             id="r1"/>
  <cust:customAction action="navigatefromview1" id="ca1"
                     location="both" icon="/Logo1.JPG"
                     text="View1 Action"
                     shortDesc="Custom View1 Action"/>
  <cust:customAction action="navigatefromview2"
                     location="both" id="ca2"
                     icon="/Logo2.JPG" text="View2 Action"
                     shortDesc="Custom View2 Action"/>
</cust:showDetailFrame>
```

Notes: You can add custom actions for all of the task flow's ADFc outcomes, but depending on the task flow view that is displayed, several or none of the custom actions are available at runtime.

If you define a custom action without a corresponding task flow action, then the custom action is not rendered on the `Show Detail Frame` header at runtime.

See Also: "Creating a Task Flow" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*

4. Save and run the page.

At runtime, when you select an action from the `Show Detail Frame`'s Actions menu, the associated control flow rule is triggered and the target task flow fragment is rendered.

22.3.2.2 Defining Custom Actions at the Global Level

Defining custom actions at the global level means making those custom actions available for all `Show Detail Frame` instances in an application. Though global-level custom actions are available on all `Show Detail Frame` components in an application, at runtime the header of any `Show Detail Frame` displays only those custom actions that correspond to the ADFc outcomes of the current view of the task flow.

Define global-level custom actions in your application's `adf-config.xml` file.

To define custom actions at the global level:

1. Open the application's `adf-config.xml` file, located in the `ADF META-INF` folder under Descriptors in the Application Resources panel.
2. Define custom actions using the `<customActions>` element with nested `<customAction>` tags for each action, as shown in Example 22–3.

   Tip: To render a custom action only in page Edit mode, you can set the `rendered` attribute on the `<customAction>` tag to `#{composerContext.inEditMode}`. This returns a value of `true` if the page is in Edit mode.

Example 22–3 Custom Actions Definitions in the adf-config.xml File

```
<cust:adf-config-child
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <enableSecurity value="true" />
  <customActions>
    <cust:customAction action="forward" displayName="Move Forward"
                       location="menu" rendered="true"
                       icon="/move_forward.png"/>
    <cust:customAction action="backward" tooltip="Move Backward"
                       location="chrome" rendered="true"
                       icon="/move_backward.png"/>
  </customActions>
</cust:adf-config-child>
```

Notes:

- If you implemented restrictions on the `Show Detail Frame` component's actions by performing the steps in Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions," you already have a `cust:customizableComponentsSecurity` section in the `adf-config.xml` file. You can define custom actions in that section itself instead of the `cust:adf-config-child` section shown in Example 22–3.
- If you are defining an icon for the custom action, you must ensure that the image you specify is available in the project root folder.
- You can define custom actions for the internal actions defined in all task flows on your page; however, at runtime, the header of any `Show Detail Frame` displays only those custom actions that correspond to the ADFc outcomes of the current view of the task flow.

3. Save the `adf-config.xml` file.

For additional information about defining custom actions, see Section 22.3.4, "How to Enable Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example."

Resolving Conflicts Between Global-Level and Instance-Level Custom Actions

Each custom action is uniquely identified by the value of its `action` attribute. If you have defined custom actions with the same `action` attribute value at the global and instance levels, then there may be conflicts in how these custom actions are invoked at runtime, depending on other attribute values. At such times, the `inheritGlobalActions` attribute of the `Show Detail Frame` defines the

behavior of other custom action attributes (other than the `action` attribute) as follows:

> **Note:** Regardless of the `inheritGlobalActions` setting (`true` or `false`) on the `Show Detail Frame` component:
>
> - The `rendered` attribute is not inherited even if it is not specified at the instance level.
> - The `location` attribute at the global or instance level should be set to the same value at both the global and instance levels.

- If `inheritGlobalActions=true`, or you have not specified a value for `inheritGlobalActions` (defaults to `false`), the behavior of custom action attributes is as follows:
  - If you defined a custom action attribute at the global and instance levels, then the attribute value specified at the instance level is used.
  - If you defined a custom action attribute only at the instance level, then that attribute value is used.
  - If you defined a custom action attribute only at the global level, then that value is ignored and the default value is used.
- If `inheritGlobalActions=true`, the behavior of custom action attributes is as follows:
  - If you defined a custom action attribute at the instance level, then its value is used regardless of whether the same attribute is specified at the global level.
  - If you defined a custom action attribute only at the global level, then that value is used.
  - If you have not defined a custom action attribute at the global or instance levels, then the attribute's default value is used.

After you have designed your application pages, you must deploy the application to the production environment. For more information, see Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application."

> **Note:** Runtime customizations that you perform on the page in the development environment are not carried over when you deploy the application to a target server.

22.3.2.3 Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window

Custom actions typically display the target task flow views in place, inside the `Show Detail Frame` component. However, you can define a custom action to display a task flow view in a separate browser window.

To display a task flow view in a separate browser window, the control flow rule to that view must be prefixed with `dialog:` in the task flow definition file and in the `action` attribute of the custom action corresponding to that view. The following example shows an `action` attribute definition:

```
<cust:customAction action="dialog:Next" id="ca1"
                   location="both" icon="/move_forward.png"
                   text="Next Action"
```

```
                    shortDesc="Next Action"/>
```

Setting Properties on the Popup Window

For a command component inside a task flow region, you can specify the default behavior using the `useWindow`, `windowEmbedStyle`, `windowHeight`, `windowWidth`, and `returnListener` attributes. The command component may have other such attributes that you can use. If you specify a return listener, on closing the dialog a particular action event is called to decide on the next navigation outcome. By default, without this setting, a dummy `Rich Command Link` component is created to trigger the task flow action.

When you define a listener on a command component, you must also configure the custom action to call the action event on this component. The `actionComponent` attribute on a custom action definition (global- and instance-level) enables you to specify the ID of the command component that must be queued for the action event. When the `actionComponent` attribute is specified, the `Show Detail Frame` component queues the action event on this component. Since this command component is inside your task flow, you change its attribute values at any time.

Example

Consider an example where you have included a task flow inside a `Show Detail Frame` component and defined a `Simple Edit` custom action corresponding to the task flow's navigation outcomes. A `Command Button` component inside the task flow is configured to launch a modal inline popup of size `300x200` on clicking the `Simple Edit` custom action. A return listener is configured on the command component so that it is called whenever the popup is closed.

The source code of the Command Button component inside the region is as follows:

```
<af:commandButton text="dialog:simpleEditPoup"
                  id="SDFCustomActionCmd_simpleEditPoup"
                  action="dialog:simpleEditPoup" useWindow="true"
                  windowEmbedStyle="inlineDocument" windowWidth="300"
                  windowHeight="200"
                  windowModalityType="applicationModal"

returnListener="#{pageFlowScope.recentPagesBean.refreshMainView}"
                  visible="false"/>
```

Example 22–4 shows how you can specify a global custom action corresponding to the task flow outcome by defining the custom action in the `adf-config.xml` file. The ID of the `Command Button` component is specified against the `actionComponent` attribute on the custom action.

Example 22–4 Global Custom Action Defined in the adf-config.xml File

```
<customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <enableSecurity value="true"/>
    <customActions>
     <customAction action="dialog:simpleEditPoup"
                   text="Simple Edit"
                   shortDesc="Simple Edit"
                   location="menu"
                   rendered="#{!changeModeBean.inEditMode}"
                   icon="/adf/pe/images/editproperties_ena.png"
                   actionComponent="SDFCustomActionCmd_simpleEditPoup"/>
    </customActions>
```

```
</customizableComponentsSecurity>
```

Example 22–5 shows how you can specify an instance-level custom action corresponding to the task flow outcome by adding a `Custom Action` component from the Composer tag library to the JSPX page. The ID of the `Command Button` component is specified against the `actionComponent` attribute on the custom action.

Example 22–5 Instance-Level Custom Action Defined in the JSPX Page

```
<cust:showDetailFrame id="sdf_for_RecentPagesTF1"
                      text="Recent Pages" stretchContent="false"
                      showResizer="never">
  <af:region id="RecentPagesTF1"
             value="#{bindings.regionBinding1.regionModel}"/>
  <cust:customAction action="dialog:simpleEditPoup"
                     text="My Simple Edit"
                     shortDesc="Simple Edit"
                     location="menu"
                     rendered="#{!changeModeBean.inEditMode}"
                     icon="/adf/pe/images/editproperties_ena.png"
                     actionComponent="SDFCustomActionCmd_simpleEditPoup"/>
</cust:showDetailFrame>
```

Note: A custom action that is launched in a popup dialog is sent as a new request to server. If you are using a Composer sandbox and are in Edit mode of a page, ensure that this request is launched in View mode by adding code in your servlet filter so that a new sandbox is not created for this page.

22.3.3 What Happens at Runtime

Custom actions configured in the `adf-settings.xml` file are merged with the custom actions configured in the `adf-config.xml` file and all actions relevant to the current view of the selected task flow are displayed on the parent `Show Detail Frame` component's Actions menu.

If you enabled custom actions at a global level, then the header of any `Show Detail Frame` displays these custom actions if they correspond to the navigation outcomes of the current view of the child task flow.

If you prefixed the `action` attribute value with `dialog:`, then the target view of the task flow opens in a separate browser window.

22.3.4 How to Enable Custom Actions On a Show Detail Frame Enclosing a Task Flow: Example

In this example, assume that your application contains a task flow, `customactions`, residing inside a `Show Detail Frame`. The task flow includes three view elements, `view_gadget`, `edit_settings`, and `about_gadget`, and three associated control flow rules, `ViewGadget`, `EditSettings`, and `AboutGadget`. Your object is to define custom actions so that the control flow rules are available as actions on the Actions menu of the `Show Detail Frame` component.

In this example, the control flow rules are added in such a way that users can navigate back and forth between the three views. Each view element has an associated page fragment of the same name:

- The `view_gadget.jsff` fragment has a `Panel Stretch Layout` component. The center facet of this component is populated with an `Active Output Text` component whose `Value` attribute is set to `View Gadget`.
- The `edit_settings.jsff` fragment has a `Panel Stretch Layout` component. The center facet of this component is populated with an `Active Output Text` component whose `Value` attribute is set to `Edit Gadget Settings`.
- The `about_gadget.jsff` fragment has a `Panel Stretch Layout` component. The center facet of this component is populated with an `Active Output Text` component whose `Value` attribute is set to `About This Gadget`.

To enable custom actions on the task flow:

1. Place the `customactions` task flow inside a `Show Detail Frame` component on a customizable page, `MyPage.jspx`.

   For information about creating a customizable page, see Section 20.1.1, "How to Create a Customizable Page."

2. Add a `Custom Action` component from the Composer tag library as a child of the `Show Detail Frame` component, and set the `Action` and `Text` attributes to `ViewGadget` and `View Gadget` respectively.
3. Add two more `Custom Action` components to the `Show Detail Frame`:
   - Set the `Action` and `Text` attributes for the first component to `EditSettings` and `Edit Settings` respectively.
   - Set the `Action` and `Text` attributes for the second component to `AboutGadget` and `About Gadget` respectively.
4. Save and run `MyPage.jspx`.

   The `view_gadget` page fragment is rendered in the `Show Detail Frame` component (named **My Gadget**) on the page. The Actions menu displays the **About Gadget** and **Edit Settings** options. Click **About Gadget** to navigate to the `about_gadget` fragment. Note that the Actions menu now displays the navigation rules for the other two fragments (Figure 22–5).

Figure 22–5 Custom Actions on a Show Detail Frame Enclosing a Task Flow



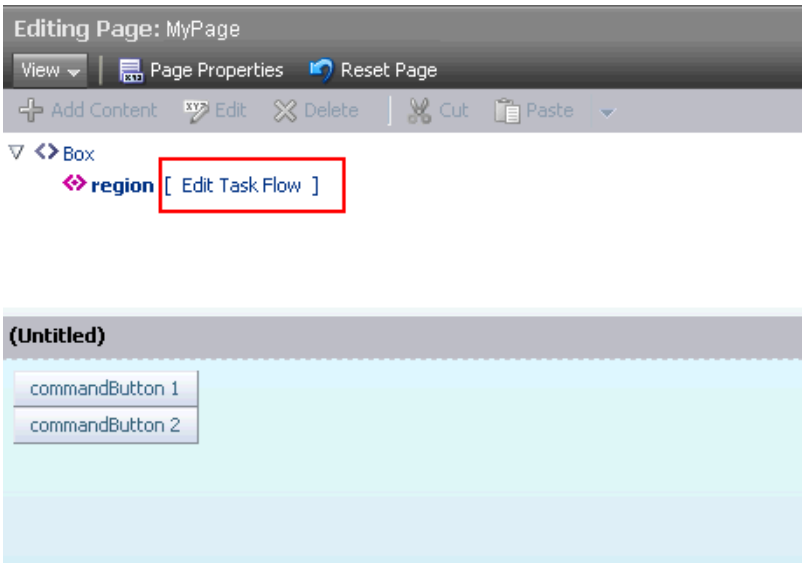
To see this in action, look at the sample application, `CustomActions.jws`, on the Oracle WebCenter Portal 11g Demonstrations and Samples page on the Oracle Technology Network (OTN):

`http://webcenter.oracle.com`

22.4 Enabling Custom Actions that Display on Task Flows in the Component Navigator

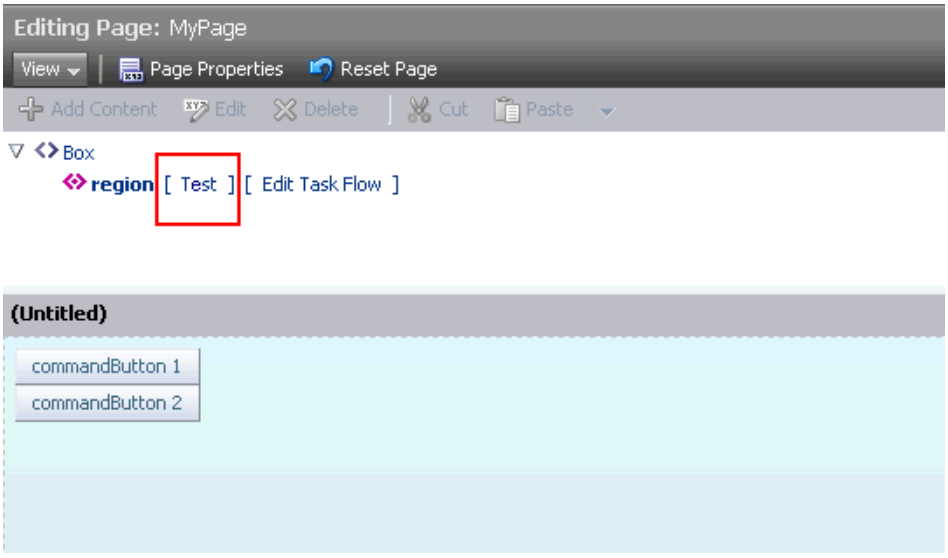
The component navigator in Composer Source view provides an option to zoom into a task flow and display the components on its page or fragment, as shown in Figure 22–6.

Figure 22–6 Edit Action on a Task Flow Instance



Users can zoom in, edit the page or fragment, and zoom out of the task flow to navigate back to the page containing the task flow. In addition to the Edit Task Flow and Close links displayed next to a task flow name, you can configure your application to also display custom actions next to a task flow name, as shown in Figure 22–7.

Figure 22–7 Custom Action on a Task Flow Instance



This section describes the procedure to enable custom actions on task flows in the component navigator. It contains the following sections:

- Section 22.4.1, "How to Configure Custom Actions in the Component Navigator"
- Section 22.4.2, "What Happens at Runtime"

22.4.1 How to Configure Custom Actions in the Component Navigator

To display custom actions against a task flow in the component navigator, you must create a Java bean that defines the custom action behavior and call this bean from the application page containing the task flow. This section describes the steps to do so in detail. It contains the following subsections:

- Section 22.4.1.1, "Defining the Logic for the Custom Action"
- Section 22.4.1.2, "Creating a JSPX Page Containing the Custom Action"
- Section 22.4.1.3, "Calling the JSPX Page from the Application Page Containing the Task Flow"

22.4.1.1 Defining the Logic for the Custom Action

To begin with, you must decide on the custom action you want to provide to users and create a Java bean with the logic to be implemented when a user selects the custom action. This section describes the steps to implement simple logic to display a message to users on clicking the custom action. The sample bean also contains the code to display the Close/Edit Task Flow links along with the custom link.

To create a Java bean:

1. From the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.
3. In the Create Java Class dialog, enter `BackingBean` in the **Name** field and click **OK**.

   The `BackingBean.java` file displays in Source view.
4. Import the required libraries as follows:

```
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import oracle.adf.view.page.editor.sourceview.ComponentInfo;
```

5. Add the following code:

```
public class BackingBean {
  public BackingBean() {
  }

  public void action(ActionEvent actionEvent) {
      FacesContext context = FacesContext.getCurrentInstance();
      context.addMessage(null,
                         new FacesMessage(FacesMessage.SEVERITY_INFO, "Sample
message to test whether the custom action works.",
                                          null));
  }

  public boolean isRendered() {
      return ComponentInfo.isRegion();
```

```
  }

  public String getZoomText() {
      return ComponentInfo.isRootNode() ? "Close" : "Edit Task Flow";
  }
}
```

With this logic, the sample message you defined is displayed when a user clicks the custom action link against a task flow region.

Use the `isRegion()` API to ensure that the custom action link is displayed only against task flow regions on the page. Use the `isRootNode()` API to ensure that the Edit Task Flow or Close link is displayed against the root component of the task flow.

6. Save the bean.

22.4.1.2 Creating a JSPX Page Containing the Custom Action

This section describes the procedure to create a JSPX page containing the custom action that you want to display to users in Composer.

To create the JSPX page:

1. In your application project, create a JSPX file called `customList.jspx`:
   a. From the **File** menu, select **New**.
   b. In the New Gallery dialog, expand **Web Tier**, select **JSF**, then **JSF Page**.
   c. Enter a name for the page and click **OK**.
2. Design the custom action UI using components like `Output Text` and `Command Link`, as shown in the following sample page:

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <jsp:directive.page contentType="text/html;charset=UTF-8"/>
  <af:componentDef var="attrs" componentVar="component">
    <af:panelGroupLayout id="dc_pgl1" rendered="#{backingBean.rendered}">
      <af:outputText value="[" id="dc_ot1"/>
      <af:commandLink text="Test" id="dc_cl1"
                      actionListener="#{backingBean.action}"
                      binding="#{backingBean.customLink}"/>
      <af:outputText value="]" id="dc_ot2"/>
      </af:panelGroupLayout>
    </af:componentDef>
</jsp:root>
```

This sample creates a custom action called `Test`. The `actionListener` and `binding` attributes on this action are bound to `BackingBean`, which you created earlier.

3. To ensure that the default Edit Task Flow/Close options are also displayed next to the task flow, you must define a facet called `zoom`, as shown in the following example:

```
<af:xmlContent>
  <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
    <facet>
```

```
        <facet-name>zoom</facet-name>
      </facet>
    </component>
  </af:xmlContent>
```

4. Include the `zoom` facet in the page content as shown in the following example:

```
<af:outputText value="[" id="dc_ot3"/>
<af:facetRef facetName="zoom"/>
<af:outputText value="]" id="dc_ot4"/>
```

The source of the `customLink.jspx` page is as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <jsp:directive.page contentType="text/html;charset=UTF-8"/>
  <af:componentDef var="attrs" componentVar="component">
    <af:panelGroupLayout id="dc_pgl1" rendered="#{backingBean.rendered}">
      <af:outputText value="[" id="dc_ot1"/>
      <af:commandLink text="Test" id="dc_cl1"
                      actionListener="#{backingBean.action}"
                      binding="#{backingBean.customLink}"/>
      <af:outputText value="]" id="dc_ot2"/>
      <af:outputText value="[" id="dc_ot3"/>
      <af:facetRef facetName="zoom"/>
      <af:outputText value="]" id="dc_ot4"/>
      </af:panelGroupLayout>
      <af:xmlContent>
        <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
          <facet>
            <facet-name>zoom</facet-name>
          </facet>
        </component>
      </af:xmlContent>
    </af:componentDef>
</jsp:root>
```

5. Save the JSPX page.

22.4.1.3 Calling the JSPX Page from the Application Page Containing the Task Flow

Let us assume that you have a simple JSPX page, `MyPage.jspx`, containing a task flow. The source of the page is as shown in the following example:

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable">
  <jsp:directive.page contentType="text/html;charset=UTF-8"/>
  <f:view>
    <af:document id="d1">
      <af:form id="f1">
        <af:panelStretchLayout topHeight="50px" id="psl1">
          <f:facet name="top">
            <pe:changeModeLink id="cml1"/>
```

```
          </f:facet>
          <f:facet name="center">
            <!-- id="af_one_column_header_stretched"  -->
            <pe:pageCustomizable id="pageCustomizable1">
              <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
                <af:region value="#{bindings.taskflowdefinition1.regionModel}"
                           id="r1"/>
              </cust:panelCustomizable>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
        </af:panelStretchLayout>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

The task flow, `taskflowdefinition1`, contains the following `view.jsff` fragment:

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <af:panelGroupLayout layout="scroll" id="pgl1">
    <af:commandButton text="commandButton 1" id="cb1"/>
    <af:commandButton text="commandButton 2" id="cb2"/>
  </af:panelGroupLayout>
</jsp:root>
```

To display the custom action that you created, you must ensure that the `customLink.jspx` page is called from within Composer. Use the `sourceViewNodeAction` attribute on the `Page Customizable` component to reference the JSPX page containing the custom action.

The `Page Customizable` tag appears as follows in the page source:

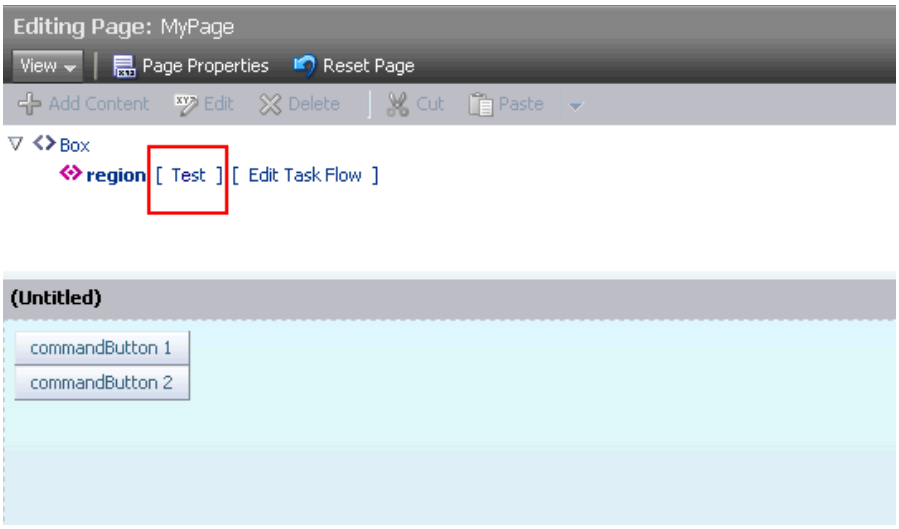
```
<pe:pageCustomizable id="pageCustomizable1"
                     sourceViewNodeAction="/customLink.jspx">
```

The `sourceViewNodeAction` attribute can take the name of a JSPX file or an EL value that evaluates to a JSPX file name.

22.4.2 What Happens at Runtime

When you run `MyPage.jspx` to the browser and open the page in Composer Source view, the component navigator displays a Test link and an Edit Task Flow link next to each task flow instance on the page. Figure 22–8 shows `MyPage.jspx` in the component navigator, with the Test link highlighted. Clicking the Test link displays a sample message to the user.

Figure 22–8 Custom Action Displayed Against a Task Flow in the Component Navigator



Composer

****

22.5 Configuring a Keyboard Shortcut to Access Composer

Typically, users enter page Edit mode by clicking the Edit link or button on the page. You can now configure your application to enable users to enter page Edit mode using keyboard shortcuts. Further, you can configure your application to run some other event on using the shortcut keys. This section explains how to enable a keyboard shortcut to Composer and configure an event handler for the shortcut keys. It includes the following subsections:

- Section 22.5.1, "How to Enable Linkless Entry Into Edit Mode"
- Section 22.5.2, "How to Configure an Event Handler for the Shortcut Key"
- Section 22.5.3, "What Happens at Runtime"

22.5.1 How to Enable Linkless Entry Into Edit Mode

You can configure linkless entry into page Edit mode by adding a `<pe:mode-switch-key>` element to the application's `adf-config.xml` file. The default keyboard shortcut configured when you set the `<pe:mode-switch-key>` element is `ctrl+shift+E`. Users can use this key sequence to toggle between page View and Edit modes. However, you can configure a key of your choice by adding a `<pe:key-sequence>` property.

To add the `<pe:mode-switch-key>` property:

1. Open the application's `adf-config.xml` file, located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the `<pe:mode-switch-key>` property and set its value to `true`, as shown in Example 22–6. The example shows the `<pe:key-sequence>` property with the shortcut value `ctrl+E`.

Example 22–6 Enabling Linkless Edit Mode Entry

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
```

```
  <pe:mode-switch-key>
    <pe:enabled>true</pe:enabled>
    <pe:key-sequence>ctrl E</pe:key-sequence>
  </pe:mode-switch-key>
. . .
</pe:page-editor-config>
```

3. Save the file.

22.5.2 How to Configure an Event Handler for the Shortcut Key

Use the `<pe:mode-switch-handler>` element to specify the event that must be triggered on using the shortcut keys, as shown in the following example:

```
<pe:mode-switch-key>
  <pe:enabled>true</pe:enabled>

<pe:mode-switch-handler>#{PageEditorBean.handleModeSwitch}</pe:mode-switch-handler
>
</pe:mode-switch-key>
```

where, `handleModeSwitch` uses the `ModeChangeEvent` method to provide the logic to toggle between two modes, as shown in the following sample:

```
public void handleModeSwitch(ModeChangeEvent editMode)
  {
    // Your implementation to handle mode change goes here
  }
```

22.5.3 What Happens at Runtime

Users can access the application page and enter Edit mode by using the configured shortcut.

22.6 Creating Event-Enabled Task Flows

Composer provides a means of contextually wiring task flow events. You can wire a contextual event to an action handler to enable the passing of values from a producer component to a consumer component when the event is triggered on the producer.

For events to be available at runtime, event capability must be included in the task flow at design time. When you add event-enabled task flows to your customizable page, each task flow's Component Properties dialog includes an Events tab, where much of the wiring activity takes place. For information about including event capabilities, see "Creating Contextual Events" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

22.7 Configuring an Application Page to Display in Source View by Default

When a user invokes Composer by clicking the Edit button or link on a page, the page opens in Design view by default. If your business so requires, you can configure your application pages to display in Composer Source view by default.

22.7.1 How to Configure an Application Page to Display in Source View by Default

The `Edit Mode view` attribute on the `Page Customizable` component enables you to select either Design view or Source view as the default page view in Edit mode.

To open a page in Source view by default:

1. Open your JSPX page in JDeveloper.
2. Select the **Page Customizable** component in the Structure Navigator.
3. In the Property Inspector, click the **Edit Mode view** attribute dropdown list and select **source**.

> **Note:** To open the page in Source view for selected users only, you can specify an EL value for the `Edit Mode view` attribute.

4. Save the page.

Setting the Source View Position and Size

By default, the component navigator in Source view is displayed at the top of the page, and its default height is `200` pixels. You can specify a different position or size for the component navigator using the `sourceViewPosition` and `sourceViewSize` attributes respectively on the `Page Customizable` component. For more information about these attributes, see Section B.1.1, "Page Customizable."

22.7.2 What Happens at Runtime

When a user clicks the Edit link or button on the page, Composer opens the page in Source view.

22.8 Disabling the Source View Menu for an Application

By default, the View menu is enabled in Composer-enabled pages, providing all users with access to Source view. You can disable the View menu if you want to prevent users from being able to edit any page components other than task flows, portlets, and layout components. This section contains the following subsections:

- Section 22.8.1, "How to Disable the Source View Menu"
- Section 22.8.2, "What Happens at Runtime"

> **Note:** For information about the editing capabilities in Source view, see Section 18.5, "Editing Capabilities in Source View in Page Edit Mode."

22.8.1 How to Disable the Source View Menu

You can disable the View menu by setting the `<pe:enable-source-view>` entry to `false` in the application's `adf-config.xml` file.

> **Note:** For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

To disable Source view:

1. Open the application's `adf-config.xml` file, located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the `enable-source-view` property and set its value to `false`, as shown in Example 22–7.

Example 22–7 Disabling Source View in adf-config.xml

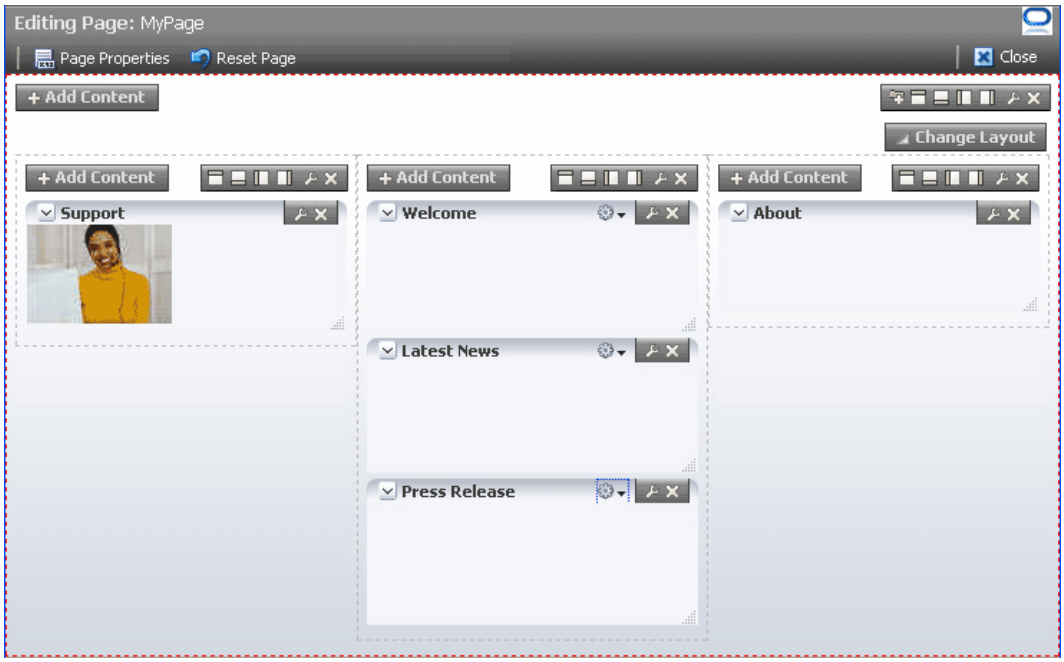
```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:enable-source-view>false</pe:enable-source-view>
</pe:page-editor-config>
```

3. Save the file.

22.8.2 What Happens at Runtime

When a user switches to page Edit mode, the page is rendered in Design view but the View menu is not displayed to the user, as shown in Figure 22–9.

Figure 22–9 Design View of Page without View Menu



Composer

****

22.9 Disabling Task Flow Zoom Capability

You can disable the capability to zoom into task flows in the application by setting the `<pe:enable-zoom>` property to `false` in the application's `adf-config.xml` file.

> **Note:** For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

To disable task flow zoom capability:

1. Open the `adf-config.xml` file, located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the `<pe:enable-zoom>` property and set its value to `false`, as shown in Example 22–8.

Example 22–8 Disabling Task Flow Zoom Capability in adf-config.xml

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:source-view>
    <pe:enable-zoom>false</pe:enable-zoom>
  </pe:source-view>
</pe:page-editor-config>
```

3. Save the file.

The ability to zoom into task flows is disabled in the entire application.

22.10 Applying Styles to Components

You can change the look and feel of Composer components by applying different styles to the component header and content using one of the following methods:

- Build a skin using style selectors, and apply the skin to a Framework application. For more information about style selectors and skins, see the chapter "Customizing the Appearance Using Styles and Skins" in *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.
- Use JDeveloper style properties to specify style information through the Property Inspector. For more information, see Understanding contentStyle and inlineStyle Properties.

> **Notes:** Using JDeveloper style properties overrides the style information from the skin CSS. However, when you define a style property using JDeveloper, this style overrides styles for the selected component only—child components continue to use the styles specified in the skin.

You can adjust the look and feel of `Page Customizable`, `Panel Customizable`, `Layout Customizable`, and `Show Detail Frame` components at design time by changing the style-related properties `inlineStyle` and `styleClass`.

`Show Detail Frame` components have another associated style property, `contentStyle`, which defines the style for the content inside the component. The `styleClass` property enables you to select an extra style from the skin, whereas the `inlineStyle` and `contentStyle` properties override the comparable styles specified in the application skin for that particular instance of the component.

The `inlineStyle` property overrides `styleClass`. Additionally, properties set on a component instance affect only that instance of the component. Other component instances in the application are not affected.

Note: The `background` property is also useful in adjusting the look and feel of `Show Detail Frame` components. It is used to provide a dark, medium, or light color scheme for the component instance. Unlike `inlineStyle`, `contentStyle`, and `styleClass` properties, the `background` property works with skins. Depending on which value is specified for a component instance's `background` property, the skin applies the relevant style.

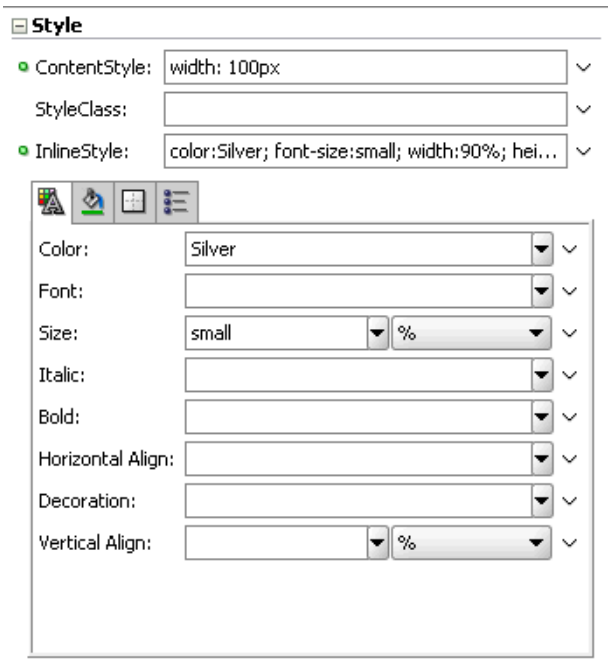
Understanding contentStyle and inlineStyle Properties

The style properties `inlineStyle` and `contentStyle` are alike in the types of attributes they support. They differ in their range of influence. While `inlineStyle` provides style information for the *entire component*, `contentStyle` provides style information only for *component content*. The `contentStyle` property is available to `Show Detail Frame` components but not to `Panel Customizable` components.

The `inlineStyle` property applies CSS to the root of the component, that is, the topmost DOM element. It does not override styles on child elements that are picking up color, font, and so on from a skin. For example, if a component header is skinned, then setting `inlineStyle` does not affect the component header. The `contentStyle` applies CSS to the DOM element that surrounds the *content* part of the component. In a `Show Detail Frame`, content refers to the area below the header.

On component content, the value specified for `contentStyle` takes precedence over the value specified for `inlineStyle`. Additionally, `contentStyle` on a component instance takes precedence over both `inlineStyle` and the `contentStyle` values of a parent component (such as a portlet nested in a `Panel Customizable` component).

Figure 22–10 Defining Styles for contentStyle and inlineStyle in the Property Inspector



22.11 Configuring the Persistence Change Manager

To persist user and application customizations across sessions, your application must be configured to use the change persistence framework. When you add Composer components to a customizable page, Composer configures the application to use `ComposerChangeManager` so that changes made to a page at runtime are persisted appropriately.

The default change persistence behavior in release 11.1.1.1 applications is to save Edit mode changes to MDS. View mode changes are not persisted and are available only for that session. To persist View mode changes also to MDS so that they are available across sessions, you can configure your 11.1.1.1 applications to use `ComposerChangeManager`.

This section describes the default change manager configuration in new Framework applications containing Composer-enabled pages and explains how to configure existing release 11.1.1.1 applications to use `ComposerChangeManager`. It contains the following sections:

- Section 22.11.1, "Overview of the Default Change Manager Configuration"
- Section 22.11.2, "How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications"

22.11.1 Overview of the Default Change Manager Configuration

The first time you add a Composer component to your application page, Composer does the following to enable change persistence:

- Adds the `CHANGE_PERSISTENCE` context parameter to the `web.xml` file, and sets the value to `ComposerChangeManager`. This context parameter registers the `ChangeManager` class to be used to handle persistence, as shown in the following example:

```
<context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>

<param-value>oracle.adf.view.page.editor.change.ComposerChangeManager</param-va
lue>
</context-param>
```

- Sets the `persistent-change-manager` element in the `adf-config.xml` file to the `MDSDocumentChangeManager`. Composer configures `MDSDocumentChangeManager` within the `adf-faces-config` section as follows:

```
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  <persistent-change-manager>
    <persistent-change-manager-class>
      oracle.adf.view.rich.change.MDSDocumentChangeManager
    </persistent-change-manager-class>
  </persistent-change-manager>
  . . .
</adf-faces-config>
```

  The `taglib-config` section in the file lists component tags and attributes that are persisted by default, as shown in the following example:

```
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
    ...
    <taglib-config>
```

```
    <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
      <tag name="showDetailFrame">
        <persist-operations>all</persist-operations>
        <attribute name="expansionMode">
          <persist-changes>true</persist-changes>
        </attribute>
        <attribute name="contentStyle">
          <persist-changes>true</persist-changes>
        </attribute>
      </tag>
      <tag name="panelCustomizable">
        <persist-operations>all</persist-operations>
      </tag>
    </taglib>
    <taglib uri="http://xmlns.oracle.com/adf/pageeditor">
      <tag name="layoutCustomizable">
        <persist-operations>all</persist-operations>
        <attribute name="type">
          <persist-changes>true</persist-changes>
        </attribute>
      </tag>
    </taglib>
  </taglib-config>
</adf-faces-config>
```

You can further enable change persistence for other tags and attributes by defining them in this section and setting the `persist-changes` attribute to `true`. For more information, see Section 24.4.1, "How to Define Change Persistence at the Component Level."

For a list of ADF Faces components and attributes that are implicitly persisted, see the chapter titled "Allowing User Customizations at Runtime" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

ComposerChangeManager

A `ChangeManager` class is required for persisting application customizations performed by end users. `ComposerChangeManager` handles change persistence both within a session and across sessions (to MDS). It delegates a users changes to the appropriate change manager as follows:

- View mode changes are routed to `FilteredPersistenceChangeManager` to ensure that implicit customizations, such as column resizing and header collapse, work according to the filter rules configured in the application's `adf-config.xml` file.
- Edit mode changes are routed to `MDSDocumentChangeManager` to ensure that both implicit and explicit customizations are always persisted and available across sessions.

Notes: When you add Composer components to your application page, the option to enable customizations for the duration of the session (in the Project Properties dialog) is disabled. This is because changes in Composer-enabled application pages must be persisted across sessions, to MDS.

22.11.2 How to Configure ComposerChangeManager in Existing 11.1.1.1 Applications

Release 11.1.1.1 Framework applications containing Composer-enabled pages use `MDSDocumentChangeManager` for persisting Edit mode changes by default. Since Composer now provides `ComposerChangeManager` to persist user and application customizations, you can configure your existing applications to use `ComposerChangeManager` so that user customizations (in View mode) are also persisted to MDS.

This section contains the following sections:

- Section 22.11.2.1, "Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File"
- Section 22.11.2.2, "Adding Relevant Entries to the adf-config.xml File"

22.11.2.1 Updating CHANGE_PERSISTENCE Context Parameter in the web.xml File

You must update the `CHANGE_PERSISTENCE` context parameter to `ComposerChangeManager`.

To modify the `CHANGE_PERSISTENCE` parameter:

1. Open your application's web.xml file from the *`Application_Root`*`\`*`Project_Name`*`\public_html\WEB-INF` directory.
2. Locate `org.apache.myfaces.trinidad.CHANGE_PERSISTENCE` and update it to the following:

```
<context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>

<param-value>oracle.adf.view.page.editor.change.ComposerChangeManager</param-va
lue>
</context-param>
```

3. Save the `web.xml` file.

22.11.2.2 Adding Relevant Entries to the adf-config.xml File

You must configure change persistence in the `adf-faces-config` section of your application's `adf-config.xml` file.

To configure change persistence in the `adf-config.xml` file:

1. Open the `adf-config.xml` file.

   The `adf-config.xml` file is located in the ADF `META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add the following code under the `adf-config` section:

```
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  <persistent-change-manager>

<persistent-change-manager-class>oracle.adf.view.rich.change.MDSDocumentChangeM
anager</persistent-change-manager-class>
  </persistent-change-manager>
  <taglib-config>
    <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
      <tag name="showDetailFrame">
        <persist-operations>all</persist-operations>
        <attribute name="expansionMode">
          <persist-changes>true</persist-changes>
```

```
        </attribute>
        <attribute name="contentStyle">
          <persist-changes>true</persist-changes>
        </attribute>
      </tag>
      <tag name="panelCustomizable">
        <persist-operations>all</persist-operations>
      </tag>
    </taglib>
    <taglib uri="http://xmlns.oracle.com/adf/pageeditor">
      <tag name="layoutCustomizable">
        <persist-operations>all</persist-operations>
        <attribute name="type">
          <persist-changes>true</persist-changes>
        </attribute>
      </tag>
    </taglib>
  </taglib-config>
</adf-faces-config>
```

This ensures that `MDSDocumentChangeManager` is used to persist View mode changes to component tags and attributes listed in the `taglib-config` section. By default, the following View mode changes are persisted:

- `Panel Customizable` component: Rearranging of components within the container or across containers.
- `Show Detail Frame` component: Expanding, collapsing, and resizing of components.

You can enable change persistence for other tags and attributes by defining them in this section and setting the `persist-changes` attribute to `true`. For more information, see Section 24.4.1, "How to Define Change Persistence at the Component Level."

3. Save the `adf-config.xml` file.

Note: When you configure `ComposerChangeManager`, implicit customizations on many components may stop working in View mode as the components may not be added to `adf-config.xml` for the `FilteredPersistenceChangeManager` to pick up.

22.12 Configuring Runtime Resource String Editing

Component properties edited in Composer are available only in the current language. For example, if you add a task flow to your page, the task flow title you enter in Composer is available only in the current language. As a result, users in different locales do not see the translated values for such properties. To provide language support for component properties edited at runtime, Composer now provides users the option to edit resource strings for component display options that can take String values. This is similar to the resource string editor feature provided by ADF Faces in JDeveloper. Changes made to resource strings in Composer are saved into an override bundle, which gets translated along with other resource bundles in the application. The translated versions are then imported back into the application. This way, users are able to see the property values in their language.

Note: Composer supports creation of around 500 resource strings at runtime. Beyond this number, application performance slows down.

Release 11.1.1.1 Framework applications containing Composer-enabled pages do not support editing of resource strings at runtime. You can enable resource string editing in your existing application by performing a few simple configurations. This section describes the steps to enable resource string editing in an existing release 11.1.1.1 application. It contains the following sections:

- Section 22.12.1, "Overview of the Resource String Editor"
- Section 22.12.2, "How to Enable the Resource String Editor in Your Application"
- Section 22.12.3, "How to Configure the Override Bundle"
- Section 22.12.4, "What Happens at Runtime"

22.12.1 Overview of the Resource String Editor

At runtime, for component display options that can take String values, the Edit menu next to the property field provides a Select Text Resource option. Clicking this option opens the resource string editor, which provides options to search existing resources, edit or delete resource keys, and add new resource key/value pairs. For information about editing resource strings at runtime, see Section 18.4.5, "Edit Resource Strings."

Changes made to resource strings in Composer are saved into an application override bundle. This bundle is sent for translation and the translated versions are imported back into the application. This way, users are able to see the property values in their language.

The `ComposerOverrideBundle.xlf` is an empty XLIFF file that is packaged with Composer libraries. This bundle becomes available to the application when you add Composer components to the page at design time. At runtime, Composer uses this bundle internally to read from and write to the application override bundle while editing resource strings.

New and updated property values are saved into the override bundle. A user must specify a key, value, and description for each string being created or modified. All resource string changes made in an application are saved in a single override bundle. To distinguish runtime changes made in different layers, the key value is prefixed with the MDS layer name and value and must be in the format, `RT_<`*`MDS Layer Name`*`><`*`MDS Layer Value`*`>_`*`Key Name`*. For example, `RT_sitewebcenter_WELCOME_MESSAGE`. For a selected property, users can search and use resource strings created in any MDS layer. However, they can edit only those resource strings that were created in the same MDS layer. In addition to searching the override bundle, users can also search for strings created and used in the JSPX page at design time. Internally, Composer searches for all resource bundles defined using `c:set` tags in the JSPX file.

Runtime resource string changes in English are saved to the default application override bundle. Users can change the language and create or edit a resource string in that language. A separate override bundle is created for each language in which a user writes a resource string and the file name is suffixed with the initials for that language. For example, resource strings edited in French in `Application1` are saved in a bundle named `Application1OverrideBundle_fr.xlf`.

Current Limitations of the Resource String Editor

- Composer supports creation of around 500 resource strings at runtime. Beyond this number, application performance slows down
- The search criteria in the resource string editor is case-sensitive.
- To search for the description of a resource string, the user must provide the complete string value in the search criteria.
- The resource string editor does not display string descriptions for entries created in Properties bundle or List Resource bundle formats in JDeveloper.
- The option to edit resource strings is disabled for properties on the Parameters and Events tabs.

  If the EL value for an ADF resource is referenced in a page definition parameter, such as page parameter, task flow parameter, or portlet parameter, then the binding EL returns an empty value. Users must avoid referencing ADF resource EL values in page definition parameters.

22.12.2 How to Enable the Resource String Editor in Your Application

To enable resource string editing in your application, you must add a `<pe:resource-string-editor>` element in the `<pe:page-editor-config>` section of your application's `adf-config.xml` file, as follows:

```
<pe:page-editor-config>
  <pe:resource-string-editor>
    <pe:enabled>true</pe:enabled>
  </pe:resource-string-editor>
</pe:page-editor-config>
```

If you want to enable resource string selectively, based on specific criteria, such as the page being edited or the user or role, you can use an EL value for the `enabled` attribute.

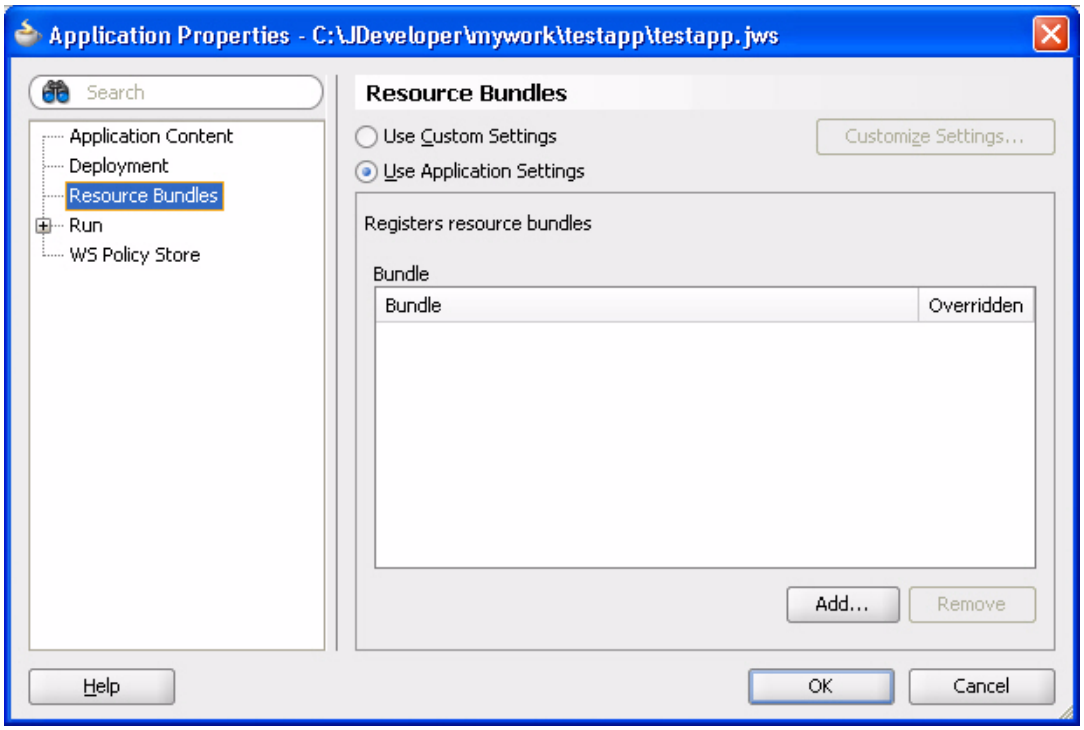
22.12.3 How to Configure the Override Bundle

If you enable resource string editing in your application, you must configure your application to use the override bundle, `oracle.adf.view.page.editor.resource.ComposerOverrideBundle.xlf`, which is available in the `pageeditor.jar` file. This override bundle is used for reading and writing to the application override bundle when users create or edit resource strings at runtime.

To configure the override bundle:

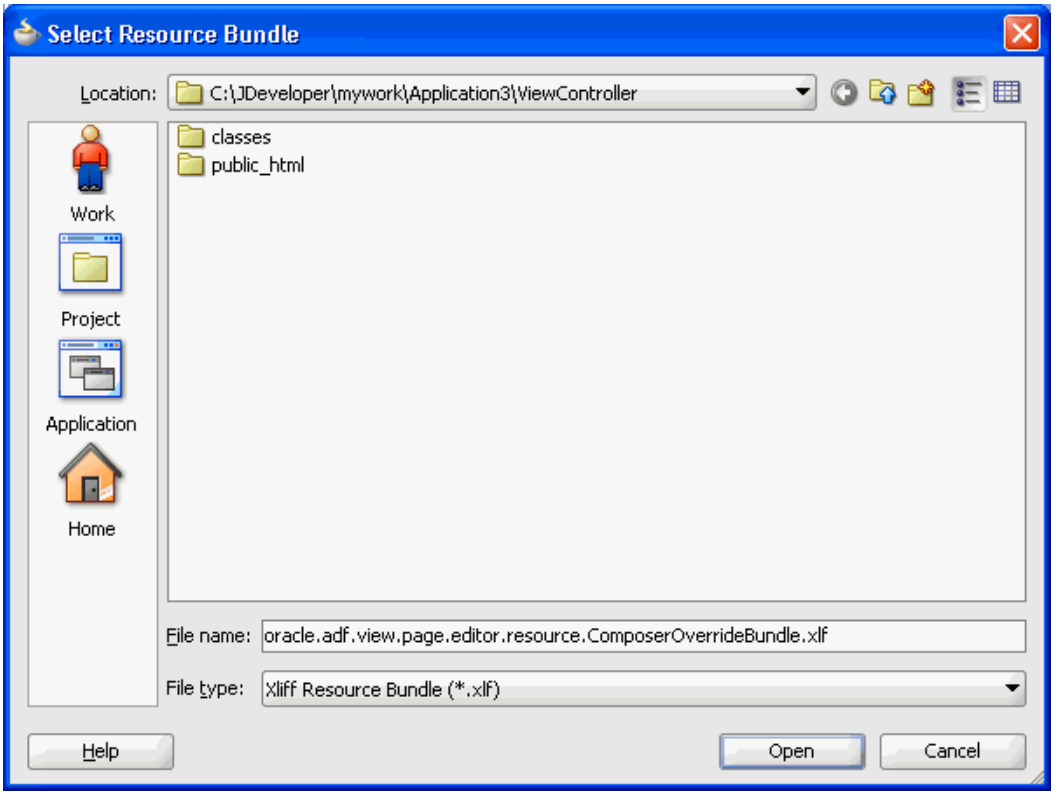
1. Open your application in JDeveloper.
2. Form the Application menu, select Application Properties.
3. Select Resource Bundles in the left panel of the Application Properties dialog (Figure 22–11).

Figure 22–11 Application Properties Dialog



4. Click the **Add** button.
5. In the File Name field in the Select Resource bundle dialog, enter `oracle.adf.view.page.editor.resource.ComposerOverrideBundle.xlf`, as shown in Figure 22–12, and click **Open**.

Figure 22–12 Select Resource Bundle Dialog



6. In the Application Properties dialog, select the **Override** check box.

   This ensures that the file is available to users at runtime.

7. Click **OK**.

Repeat the steps in this section to expose any of the resource bundles in your application to users at runtime.

> **Note:** To make a resource bundle accessible for runtime read and write operations, you must ensure that the Override option is selected for that bundle in the Application Properties dialog.
>
> You can make properties bundles, list resource bundles, and XLIFF resource bundles accessible at runtime.

22.12.4 What Happens at Runtime

When a user clicks the **Edit** icon next to a property that takes a String value, the Select Text Resource option is available. Clicking this option opens the resource string editor in which the user can create, modify, or delete resource strings. For more information, see Section 18.4.5, "Edit Resource Strings."

22.13 Troubleshooting Problems with Advanced Composer Configurations

This section provides information to assist you in troubleshooting problems you may encounter on performing advanced Composer configurations.

For information about configuring logging, see "Configuring ADF Logging for Composer".

Problem

You added a global- or instance-level `Custom Action`. However, it neither displays on the chrome or in the **Action** menu on the `Show Detail Frame`.

Solution

Ensure the following:

- The first child of the `Show Detail Frame` must be `af:region`.
- The view currently displayed on the task flow must have an outcome in its task flow definition file that matches the action of `Custom Action`.
- If the action has the prefix `dialog:`, the outcome in the task flow definition must also have the same prefix.

For information, see Section 22.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow."

Problem

Composer is not working on ADF application pages. Errors are reported in the logs when using WebCenter Portal's Resource Catalog or Component Properties dialog. None of the customizations are saved. Objects are not getting added from the catalog.

Solution

Composer is compatible with the `MDSDocumentChangeManager` and `ComposerChangeManager`. You must set the `CHANGE_PERSISTENCE` context parameter in the `web.xml` file to either of these. The following example shows how to set CHANGE_PERSISTENCE to `MDSDocumentChangeManager`:

```
<context-param>
 <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>
 <param-value>oracle.adf.view.rich.change.MDSDocumentChangeManager</param-value>
</context-param>
```

When you configure Composer, the value of `CHANGE_PERSISTENCE` is changed to `oracle.adf.view.page.editor.change.ComposerChangeManager`.

23

Performing Composer-Specific MDS Configurations

This chapter describes how to create customization layers for saving changes made to the page at runtime. It also describes how to enable Composer sandbox creation to provide users the option to preview their changes before saving them to the back end. It contains the following sections:

- Section 23.1, "Introduction to MDS"
- Section 23.2, "Using Composer Sandbox"
- Section 23.3, "Adding Customization Layers to View and Edit Modes: Example"

23.1 Introduction to MDS

Most industries customize their enterprise applications to serve different audiences and domains. Problems can arise when an application is modified at the site level. For example, upgrading an application with site-level customizations may lead to data loss or data-merge errors. Consequently, a new version of the application cannot be deployed until all merge conflicts are reconciled.

In the metadata domain, MDS provides the customization feature to address such problems. The customization feature allows for the creation of nonintrusive customization layers that are applied on top of the base application definitions. Customization layers, or layered changes, are described in their own documents and are stored separately from the base application definition. At runtime, applicable customizations are loaded from the metadata store and layered over the base metadata definition to produce the desired effect. Product upgrades and patches affect only the base metadata definition, so customizations continue to function properly.

Customization Layers

MDS enables clients to specify multiple customization types. For example, you can add customizations to runtime modes, application or user roles, application states, or any client specified criteria. Each such customization type is called a *customization layer* and is depicted using a `CustomizationClass`. A `CustomizationClass` is the interface MDS uses to identify the customization layer to be overlaid on the base definition. For example, you can configure your application to save customizations based on the departments to which users belong. You can create a customization layer called `DepartmentCC` in which application customizations made by users in different departments are stored in different folders within the layer. Another example would be to configure MDS to save View mode changes as user customizations in one layer and Edit mode changes as application customizations in another layer or vice versa.

Customization layers are applied in order of precedence, that is, if the same change is made in two different layers that apply to the given user and session, the change defined in the higher precedence layer is applied first.

When you implement a `CustomizationClass`, you must register it with the MDS. The MDS provides a means of associating a list of `CustomizationClass` types with a single `MetadataObject`. This is called the *fine-grained* association. The MDS also provides the means of associating a list of `CustomizationClass` types with a set of `MetadataObjects`. This is called the *coarse-grained* association. For information about the `CustomizationClass` and about creating a customization layer, see Section 23.3, "Adding Customization Layers to View and Edit Modes: Example."

Composer Sandbox

Typically, in a Framework application, runtime customizations are saved immediately in the *JDEV_HOME*/jdev/*system_directory*/o.mds.dt/adrs/*application_name*/AutoGeneratedMar/mds_adrs_writedir directory. Changes made in both View and Edit modes are saved in this way. However, in certain circumstances, users might first want to apply customizations in their own view and evaluate whether to keep or cancel the changes before actually saving them to the back end. You can configure Composer to create a *sandbox* if you are using a database repository to store customizations. For information about creating a sandbox, see Section 23.2, "Using Composer Sandbox."

23.2 Using Composer Sandbox

A sandbox is a temporary storage layer for saving runtime page customizations until they are committed to the back end. If you configure a sandbox, a Save button is displayed on the Composer toolbar to enable users to save their changes. In a sandbox-enabled application, if a user clicks Close without first saving changes, a Close Confirm dialog prompts the user to save or cancel changes before closing Composer.

User customizations made in View mode are saved immediately. As such changes are available only to the user modifying the page, there is no particular value in reviewing such changes before saving.

Since Edit mode customizations are available to all users who access the page, you can enable a sandbox so that a user can experiment with page customizations and assess them before committing them. If you enable a sandbox for the application, a Save button is displayed on the Composer toolbar in both Design view and Source view of the page.

This section discusses the steps you can take to enable sandbox creation and describes runtime behavior of a sandbox-enabled application. It contains the following subsections:

- Section 23.2.1, "How to Enable Composer Sandbox Creation"
- Section 23.2.2, "What Happens at Runtime"
- Section 23.2.3, "How to Disable Sandbox for an Application"
- Section 23.2.4, "How to Destroy Stale Sandboxes"
- Section 23.2.5, "How to Enable Application Sandbox Creation"
- Section 23.2.6, "What Happens When You Enable Application Sandbox Creation"

23.2.1 How to Enable Composer Sandbox Creation

You can enable a sandbox only if your application uses a database store. Therefore, you must ensure that you have configured a database store before performing the steps in this section. For information about setting up a database store, see *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

This section describes how to enable the creation of a Composer sandbox. It contains the following subsections:

- Section 23.2.1.1, "Updating Your Application's adf-config.xml File"
- Section 23.2.1.2, "Updating Your Application's web.xml File"
- Section 23.2.1.3, "Selecting a Database Store for Your Application"
- Section 23.2.1.4, "Enabling a Full Page Refresh at Runtime"

23.2.1.1 Updating Your Application's adf-config.xml File

By default, sandbox creation is enabled for all namespaces that are mapped to the default customization store (defined in the `<mds-config>` section of the `adf-config.xml` file). You can, however, enable sandbox for a selected number of namespaces only. To do this, you must add a `<pe:sandbox-namespaces>` element to the `adf-config.xml` file, and then define the namespaces for which you want to support sandbox creation.

> **Note:** For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

To configure sandbox creation in `adf-config.xml`:

1. Open the `adf-config.xml` file located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Under the `<metadata-namespaces>` element, ensure that `<namespace>` elements are defined for all metadata for which you want to enable sandbox creation, as shown in Example 23–1.

Example 23–1 Namespace Definitions in the adf-config.xml File

```
<!-- Your jspx customizations alone go here -->
   <namespace path="/pages" metadata-store-usage="WebCenterFileMetadataStore">
     <namespace-restriction type="CUSTOMIZATIONS"/>
   </namespace>
<!-- Your pagedef customizations alone go here -->
   <namespace path="/pageDefs" metadata-store-usage="WebCenterFileMetadataStore">
     <namespace-restriction type="CUSTOMIZATIONS"/>
     </namespace>
```

3. Add the `<pe:sandbox-namespaces>` element within the `<pe:page-editor-config>` section and add individual `<pe:namespace>` tags for all namespaces for which you want to enable sandbox creation, as show in Example 23–2.

Example 23–2 Configuring Namespaces for Sandbox Creation

```
<pe:sandbox-namespaces>
```

```
  <pe:namespace path="/pages"/>
  <pe:namespace path="/pageDefs"/>
</pe:sandbox-namespaces>
```

4. Save the `adf-config.xml` file.

23.2.1.2 Updating Your Application's web.xml File

To ensure that a sandbox is configured when you are in Edit mode of a page, you must create a filter in your application's `web.xml` file and set the appropriate filter mappings. All requests are then routed through this filter, and a sandbox is created for all Edit mode customizations. If you are using a file system metadata store, then there is no action performed on a filtered request.

> **Note:** For information about the Composer-specific configurations you can make in `web.xml`, see Section B.2.4, "web.xml."

This section provides the example of defining a Composer-specific filter and its relevant filter mappings in your application's `web.xml` file. It describes how to add the filter, `WebCenterComposerFilter`.

To define a Composer-specific filter and the filter mappings:

1. Open the *`application_root`*`/Portal/public_html/WEB-INF/web.xml` file.
2. Add `WebCenterComposerFilter` before the `adfBindings` filter:

```
<filter>
  <filter-name>composerFilter</filter-name>

<filter-class>oracle.adf.view.page.editor.webapp.WebCenterComposerFilter</filte
r-class>
</filter>
```

3. Add corresponding `<filter-mapping>` elements before the filter mapping for `AdfBindingFilter` as shown in Example 23–3.

Example 23–3 Filter Mappings for Composer-Specific Filter

```
<filter-mapping>
  <filter-name>composerFilter</filter-name>
  <servlet-name>Faces Servlet</servlet-name>
  <dispatcher>FORWARD</dispatcher>
  <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

The `<url-pattern>` tag is used to specify the pages that must pass through the `WebCenterComposerFilter` so that sandbox creation can be enabled when a page goes into Edit mode.

> **Note:** The order in which you define filters is important. Requests pass through these filters sequentially and if they are not defined in the correct order, the application will not run as designed. Your `web.xml` file must have the `WebCenterComposerFilter` implementation first, followed by the `AdfBindingFilter` implementation.

4. Save the `web.xml` file.

23.2.1.3 Selecting a Database Store for Your Application

To enable sandbox creation, your application must be configured to use a database store. Therefore, while deploying your application to a WebLogic Managed Server, ensure that you select a database repository and specify a partition name for the application.

For information about deploying your application from JDeveloper to a WebLogic Managed Server, see Section 70.3.4.1, "Deploying to a Managed Server Using Local Data Sources."

23.2.1.4 Enabling a Full Page Refresh at Runtime

When you enable sandbox creation in your application, you must ensure that a full page refresh is performed on entering and exiting page Edit mode. For the detailed steps to be performed, see Section 23.3.7, "How to Redirect the Servlet to Enable Switch Between MDS Customization Layers."

23.2.2 What Happens at Runtime

When a user switches to page Edit mode, the presence of a **Save** button on the Composer toolbar indicates that sandbox creation is enabled for the application. Changes made to the page or shared components on the page are not saved until the user clicks Save. Once saved, the customizations are available to all application users. If the **Save** button is not rendered, then sandbox is not available and each change is committed immediately.

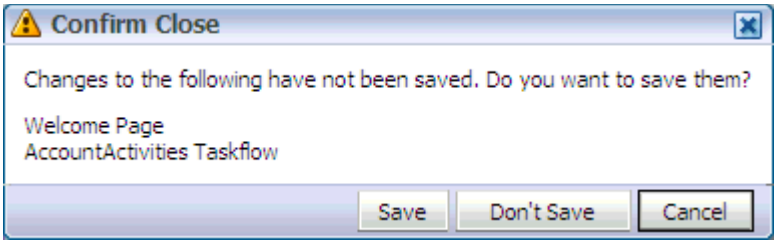
When editing component properties in the Component Properties dialog, clicking **Apply** results in the following:

- If the sandbox is not available, then the page is refreshed to display the changes made to component properties and changes are saved to the back end.
- If the sandbox is available, then the page is refreshed to display the changes made to component properties. To save changes, user must click **Save**.

Clicking **Save** or **Close** on the page results in the following events:

- On clicking **Save**, the sandbox is committed and a new sandbox is created. The page remains in Edit mode.
- On clicking **Close**, either of the following two events can occur:
  - If there are no changes since the last Save operation, then the sandbox is destroyed and Composer is closed, taking the user back to View mode.
  - If there are unsaved changes, then a Close Confirmation dialog lists the page name and unsaved task flow names, as shown in Figure 23–1.

Figure 23–1 Confirm Close Dialog in Composer



Users can select from the following options:

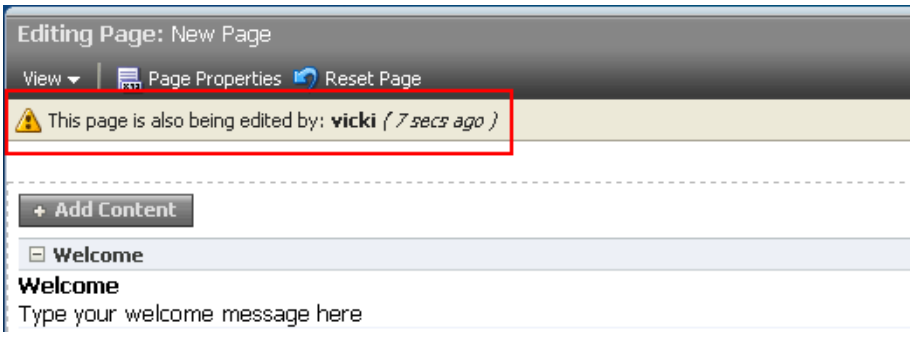
* Click **Save** to commit the sandbox and close Composer.
* Click **Don't Save** to destroy the sandbox and close Composer.
* Click **Cancel** to close the dialog and return to Composer without saving the changes.

Notes: If a user navigates away from a page while editing it and then returns to the page, any unsaved changes are lost.

What Happens During Concurrent Edits

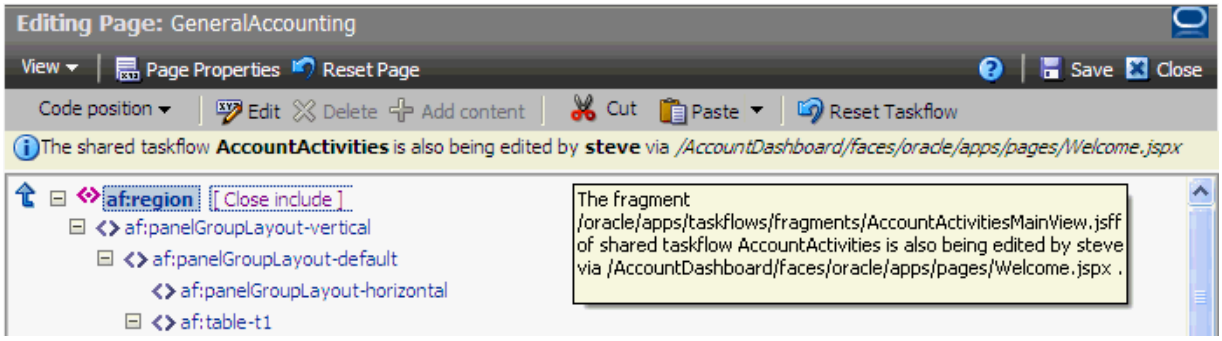
If two or more users are editing the same page or task flow using the same customization layer, then the page displays a message to each user that another user is editing the page or task flow, as shown in Figure 23–2.

Figure 23–2 Page Concurrency Message



If two or more users have zoomed into a task flow at the same time in the same customization layer and are editing it, then a concurrency message appears, as shown in Figure 23–3.

Figure 23–3 Task Flow Concurrency Message



Note: If a user zoomed into a task flow, made some customizations, and zoomed out of the task flow, then the task flow concurrency message continues to display to other users until that user saves the customizations.

Changes that are saved last overwrite prior changes. For example, if users A and B are editing a page or task flow simultaneously, then concurrency issues are handled as follows:

- If A saves the page or task flow first, then A's changes are committed to MDS. Later, when B saves the page or task flow, A's changes are overwritten with B's changes.
- If A deletes a component while B is trying to personalize (say move) that same component in View mode, a WebCenter error page is displayed to B. B has to simply navigate back to the original page. The deleted component does not appear, and B can continue working on other components.

23.2.3 How to Disable Sandbox for an Application

To ensure that changes are committed to the back end immediately, you can disable the sandbox.

To disable sandbox:

1. Open the `adf-config.xml` file located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Under the `<pe:page-editor-config>` element, add the `<pe:disable-sandbox>` attribute and set it to `true`.
3. Save `adf-config.xml`.

23.2.4 How to Destroy Stale Sandboxes

When users edit a page at runtime, if the browser closes unexpectedly or the user navigates away from the page, the sandbox used in that session is still available. Any other user accessing the same page continues to see a concurrency message that another user is editing the same page. You can destroy such stale sandboxes to free some space in the database store and enhance performance.

You can ensure that a stale sandbox is destroyed when:

- A session times out.

  The `<session-timeout>` element in the application's `web.xml` file defines the time duration after which the sandbox is destroyed. To ensure this happens, you must configure `WebCenterComposerSessionListener` in your application's `web.xml` file. This listener is called when a session times out. It creates a new sandbox when the same user enters page Edit mode.

- The user logs in to Edit mode of the page again, with the same user name.

  In this case, a new sandbox is created when the same user switches to Edit mode.

To configure `WebCenterComposerSessionListener`:

1. Open the *`application_root`*`/Portal/public_html/WEB-INF/web.xml` file.
2. Define a new listener, `WebCenterComposerSessionListener`, as shown in the following example:

```
<listener>
  <description>Composer Http Session Listener</description>
  <display-name>Composer Http Session Listener</display-name>
  <listener-class>
    oracle.adf.view.page.editor.webapp.WebCenterComposerSessionListener
```

```
    </listener-class>
  </listener>
```

3. Save the `web.xml` file.

For information about the Composer-specific configurations you can make in `web.xml`, see Section B.2.4, "web.xml."

23.2.5 How to Enable Application Sandbox Creation

Consider an example where you have created an application that displays a banner at the top of the page, a sidebar on the left with options to manage pages, and the page at the center of the browser window. You may want to enable users to edit the page in Composer and edit the sidebar using some other mechanism such as updating an XML file. Enabling Composer sandbox in such a case only saves changes to page content but not the sidebar. If you enable the application sandbox, then all changes made at runtime are saved at one time, to one location. If you enable sandbox creation at the application level, then Composer does not create its own sandbox when page Edit mode is invoked. Instead, the application sandbox is used. That is, a user interface provided by the application is used for committing and destroying the sandbox. To enable the application to provide a sandbox for Composer, you must first perform a set of configurations.

Another example where it is useful to enable the application sandbox is if you want users to edit page content, but ensure that the page is saved only on approval from the manager. By using an application sandbox, changes made by a user are in the sandbox until the manager approves the changes. The page is saved on approval.

To enable an application sandbox for use in Composer:

1. Implement a `ComposerSessionOptionsFactory` class to provide `MDSSessionOptions` for each request.

   For information about performing this task, see Section 23.3.4, "How to Implement the ComposerSessionOptionsFactory Class."

2. Register your implementation with Composer.

   For information about performing this task, see Section 23.3.5, "How to Register the Implementation with Composer."

3. Configure `WebCenterComposerFilter`.

   For information about performing this task, Section 23.3.6, "How to Configure WebCenterComposerFilter."

23.2.6 What Happens When You Enable Application Sandbox Creation

At runtime, the application-defined user interface enables users to save their changes to the back end or close Composer without saving the changes. In this case, Composer has no control over when customizations are canceled or committed to the back end.

23.3 Adding Customization Layers to View and Edit Modes: Example

You can apply customizations to a metadata object based on client-defined criteria. For example, you can customize an application and the metadata objects that it uses based on an end user's permissions, an application's deployment location (also called *localization*), or a specific industry domain. Each such category—permissions, localization, and domain—denotes a customization layer, and each is depicted using a `CustomizationClass`. A `CustomizationClass` is the interface MDS uses to

identify the customization layer to be overlaid on the base definition. See Section 23.3.3, "How to Create a Custom UserCC Tip Layer" for an example.

When you implement a `CustomizationClass`, you must also register it with the MDS. The MDS provides the ability to associate a list of `CustomizationClass` types with a single `MetadataObject`. This is called *fine-grained* association. The MDS also provides the ability to associate a list of `CustomizationClass` types with a set of `MetadataObjects`. This is called the *coarse-grained* association.

A customizable application can have multiple customization layers. You can select the layer to which you want to apply customizations. The layer you choose to customize is called the tip layer. When you drop Composer components onto a JSF page, the ADF configures a default `SiteCC` (`site`) tip layer in the application. Consequently, the `adf-config.xml` file is updated to include the `SiteCC` customization class. This tip layer stores all customizations made to the page.

This section explains through example how adding customization tip layers to View and Edit runtime modes provides user customization capabilities to all users and application customization capabilities to selected users. To enable application customizations in the Edit mode, the `site` tip layer is added. To enable user customization in the View mode, the `user` tip layer is added. By default, the `user` tip layer is applied on top of the `site` tip layer. The `user` tip layer stores all user customizations made in the View mode in a specific location created for the user who made them. Such changes are visible only to that user. The `site` tip layer stores all application customizations made in the Edit mode and are visible to all users.

To enable tip layers at runtime, Composer provides the `WebCenterComposerFilter` filter and supplies a means of defining an abstract factory for creating the MDS `SessionOptions` object. This object provides applicable customization layers at runtime and enables users to perform user customizations (View mode) or application customizations (Edit mode) based on their role. When creating a new MDS session, the `MDSSession.createSession` method of this object is used to specify the session options.

This section provides an example exercise for creating, implementing, and registering customization layers, configuring `WebCenterComposerFilter`, and switching between MDS customization layers. It contains the following subsections:

- Section 23.3.1, "How to Add Composer to a JSF Page"
- Section 23.3.2, "How to Create a Custom SiteCC Tip Layer"
- Section 23.3.3, "How to Create a Custom UserCC Tip Layer"
- Section 23.3.4, "How to Implement the ComposerSessionOptionsFactory Class"
- Section 23.3.5, "How to Register the Implementation with Composer"
- Section 23.3.6, "How to Configure WebCenterComposerFilter"
- Section 23.3.7, "How to Redirect the Servlet to Enable Switch Between MDS Customization Layers"
- Section 23.3.8, "What Happens at Runtime"

23.3.1 How to Add Composer to a JSF Page

This section describes how to add the `Page Customizable` component to a JSF page. The purpose of this exercise is to provide Composer in the Edit mode at runtime so that the `admin` user can perform customizations at the site level. This section includes the addition of a `Change Mode Link` to enable switching from View mode to Edit mode at runtime.

To add Composer to a JSF Page:

1. Open the JSPX page and select **Composer** from the Component Palette.
2. Select **Change Mode Link** and drop it onto the page.
3. Select **Page Customizable** and drop it onto the page.

   The Source view should like this:

```
<af:form id="f1">
  <pe:changeModeLink id="cml1"/>
  <pe:pageCustomizable id="pageCustomizable1">
    <cust:panelCustomizable id="panelCustomizable1" layout="scroll"/>
      <f:facet name="editor">
        <pe:pageEditor id="pep1"/>
      </f:facet>
  </pe:pageCustomizable>
</af:form>
```

   When you drop these components onto the page, the default `SiteCC` tip layer is extended by the ADF. Consequently, the `adf-config.xml` file is updated with this customization class:

```
<cust-config>
  <match>
    <customization-class name="oracle.adf.share.config.SiteCC" />
  </match>
</cust-config>
```

> **Note:** If you want your application to use `UserCC` as the default tip layer, you can simply replace `oracle.adf.share.config.SiteCC` with `oracle.adf.share.config.UserCC` in the `cust-config` section of the `adf-config.xml` file. However, to use `UserCC` as the default tip layer, you must have secured your application. You cannot implement a `UserCC` tip layer without provisioning users and roles in your application.

23.3.2 How to Create a Custom SiteCC Tip Layer

This section describes how to create a custom `SiteCC` tip layer, `site`, in which all site-level customizations performed in Edit mode are stored. In this sample application, site-level customizations are stored in the `/mds/mdssys/cust/site/webcenter/`*`pagename`*`.jspx.xml`directory.

To create the `site` tip layer:

1. From the File menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.
3. In the **Name** field, enter `SiteCC`.
4. In the **Extends** field, enter `CustomizationClass`.

   This imports `oracle.mds.cust.CustomizationClass`.
5. Click **OK.**

   The `SiteCC.java` file is rendered in the Source view.

6. To implement the `getCacheHint`, `getName`, and `getValue` methods, double-click the bulb on the left corner and select **Implement Methods**.
7. In the Implement Methods dialog, click **OK**.
8. In the Source view, press `Enter` after `public class SiteCC extends CustomizationClass`, and add the following `string`:

```
{
  private static final String DEFAULT_LAYER_NAME = "site";
  private String mLayerName = DEFAULT_LAYER_NAME;
  private String mLayerValue = "webcenter";
```

The following libraries are imported:

```
import oracle.mds.core.MetadataObject;
import oracle.mds.core.RestrictedSession;
import oracle.mds.cust.CacheHint;
```

9. Update the code as shown in **bold** here:

```
public class SiteCC extends CustomizationClass
{
  private static final String DEFAULT_LAYER_NAME = "site";
  private String mLayerName = DEFAULT_LAYER_NAME;
  private String mLayerValue = "webcenter";
  //Note: You can provide any site name.

  public CacheHint getCacheHint()
  {
    return CacheHint.ALL_USERS;
  }

  public String getName()
  {
    return mLayerName;
  }

  public String[] getValue(RestrictedSession mdsSession, MetadataObject mo)
  {
    return new String[] { mLayerValue };
  }
}
```

10. Save the `SiteCC.java` file.

23.3.3 How to Create a Custom UserCC Tip Layer

This section describes an example showing how to create a custom `user` tip layer for a user, `scott`. This layer is applied on top of the `site` layer. The user customizations that a user performs in View mode are saved in this `user` tip layer in a folder created specifically for the logged-in user. In this example, the user customizations performed in View mode are saved in the `/mds/mdssys/cust/user/scott/`*`pagename`*`.jspx.xml`directory.

> **Note:** For illustration purpose, this example describes a simple scenario where a `UserCC` tip layer is created for just one user, `scott`. In a real life application, you can configure the `UserCC` layer to return the name of the user who requested the page.

To create the user tip layer:

1. From the File menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.
3. In the **Name** field, enter `UserCC`.
4. In the **Extends** field, enter `CustomizationClass`.

   This imports the `oracle.mds.cust.CustomizationClass`.
5. Click **OK.**

   The `UserCC.java` file displays in the Source view.
6. To implement the `getCacheHint`, `getName`, and `getValue` methods, double-click the bulb on the left corner and select **Implement Methods**.
7. In the Implement Methods dialog, click **OK**.
8. In the Source view, press `Enter` after `public class UserCC extends CustomizationClass`, and add the following `string`:

```
{
  private static final String DEFAULT_LAYER_NAME = "user";
  private String mLayerName = DEFAULT_LAYER_NAME;
  private String mLayerValue = "scott"; //The name of the logged-in user
in this example. The logged-in user name can be acquired dynamically.
```

   The following libraries are imported:

```
import oracle.mds.core.MetadataObject;
import oracle.mds.core.RestrictedSession;
import oracle.mds.cust.CacheHint;
```

9. Update the code as shown in **bold** here:

```
public class UserCC extends CustomizationClass
{
  private static final String DEFAULT_LAYER_NAME = "user";
  private String mLayerName = DEFAULT_LAYER_NAME;
  private String mLayerValue = "scott";

  public CacheHint getCacheHint()
  {
    return CacheHint.USER;
  }

  public String getName()
  {
    return mLayerName;
  }

  public String[] getValue(RestrictedSession mdsSession, MetadataObject mo)
  {
    return new String[]{mLayerValue};
  }
}
```

10. Save the `UserCC.java` file.

23.3.4 How to Implement the ComposerSessionOptionsFactory Class

In this section, the `ComposerSessionOptionsFactory` class provided by the ADF is implemented to supply MDS `SessionOptions` for each HTTP request.

To implement the `ComposerSessionOptionsFactory` class:

1. From the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.
3. In the **Name** field, enter `AppsSessionOptionsFactoryImpl`.
4. Click **OK.**

   The `AppsSessionOptionsFactoryImpl.java` file is rendered in the Source view.
5. Import the following libraries:

```
import oracle.adf.view.page.editor.mds.ComposerSessionOptionsFactory;
import oracle.adf.view.page.editor.mode.ModeContext;
import oracle.mds.config.CustClassListMapping;
import oracle.mds.config.CustConfig;
import oracle.mds.core.SessionOptions;
import oracle.mds.cust.CustClassList;
import oracle.mds.cust.CustomizationClass;
```

6. Add the following code to implement the `ComposerSessionOptionsFactory` class and provide `SessionOptions`:

```
public class AppsSessionOptionsFactoryImpl
  implements ComposerSessionOptionsFactory
{

  public SessionOptions createSessionOptions(SessionOptions
defaultSessionOptions, String mode)
  {
    CustomizationClass[] custLayer;
    CustConfig custConfig = null;

    if (ModeContext.EDIT_MODE.equals(mode))
    {
      //Mode is Edit, change to SiteCC
      custLayer = EDIT_LAYER;
    }
    else
    {
      //Mode is View, change to UserCC + SiteCC
      custLayer = VIEW_LAYER;
    }

    try
    {
      CustClassList custClassList = new CustClassList(custLayer);
      CustClassListMapping custClassListMapping =
        new CustClassListMapping("/", null, null, custClassList);
      custConfig = new CustConfig(new CustClassListMapping[]
        { custClassListMapping });
    }
    catch (Exception e)
```

```
    {
      e.printStackTrace();

    }
if(defaultSessionOptions.getServletContextAsObject() != null)
{
    return new SessionOptions(defaultSessionOptions.getIsolationLevel(),
                              defaultSessionOptions.getLocale(), custConfig,
                              defaultSessionOptions.getVersionContext(),
                              defaultSessionOptions.getVersionCreatorName(),
                              defaultSessionOptions.getCustomizationPolicy(),

defaultSessionOptions.getServletContextAsObject());
  }
    else
    {
      return new SessionOptions(defaultSessionOptions.getIsolationLevel(),
                              defaultSessionOptions.getLocale(), custConfig,
                              defaultSessionOptions.getVersionContext(),
                              defaultSessionOptions.getVersionCreatorName(),
                              defaultSessionOptions.getCustomizationPolicy());
    }
  }

  //Edit mode SiteCC
  private static final CustomizationClass[] EDIT_LAYER =
    new CustomizationClass[]
    { new SiteCC() };

  //View mode SiteCC + USerCC
  private static final CustomizationClass[] VIEW_LAYER =
    new CustomizationClass[]
    { new SiteCC(), new UserCC() };

}
```

23.3.5 How to Register the Implementation with Composer

For the `site` and `user` customization layers to function, you must register the `ComposerSessionOptionsFactory` class with Composer. For example, if the concrete class is `view.AppsSessionOptionsFactoryImpl`, the following snippet must be added to the `adf-config.xml` file located in the `/.adf/META-INF` folder in your application directory:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">

<pe:session-options-factory>view.AppsSessionOptionsFactoryImpl</pe:session-options
-factory>
</pe:page-editor-config>
```

23.3.6 How to Configure WebCenterComposerFilter

You must configure the `WebCenterComposerFilter` filter in the `web.xml` file located in the `Portal/public_html/WEB-INF` folder in your application directory. This filter registers Composer's concrete `SessionOptionsFactory` with the ADF for every HTTP request. When the filter receives a call from the ADF, it forwards the request to the application and gets the `SessionOptions` with the new customized

layer. If you have not set the `Sandbox` or `VersionContext` in the `SessionOptions`, then Composer sets its own Sandbox and returns it to the ADF. For more information on Sandbox, see Section 23.2.1, "How to Enable Composer Sandbox Creation." The `composerFilter` and its filter mapping must be configured before `ADFBindingFilter`. For example, see the following `web.xml` file:

```
....

  <!-- WebCenterComposerFilter goes here -->
  <filter>
    <filter-name>composerFilter</filter-name>

<filter-class>oracle.adf.view.page.editor.webapp.WebCenterComposerFilter</filter-c
lass>
  </filter>
  <filter>
    <filter-name>adfBindings</filter-name>
    <filter-class>oracle.adf.model.servlet.ADFBindingFilter</filter-class>
  </filter>
.....
  <!-- WebCenterComposerFilter mapping goes here -->
  <filter-mapping>
    <filter-name>composerFilter</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <filter-mapping>
    <filter-name>adfBindings</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
....
```

Note: The order in which you define filters is important. Requests pass through these filters sequentially and if they are not defined in the correct order, the application will not run as designed. Your `web.xml` file must have the `WebCenterComposerFilter` implementation first, followed by the `AdfBindingFilter` implementation.

For information about the Composer-specific configurations you can make in `web.xml`, see Section B.2.4, "web.xml."

23.3.7 How to Redirect the Servlet to Enable Switch Between MDS Customization Layers

To redirect the servlet, that is, to refresh the full page at runtime, you must create:

- The `AppNavigationUtils` class, which calls the `AppNavigationUtils.redirectToSamePage()` method
- The `AppCloseHandler` CloseListener, which uses the `AppNavigationUtils` class
- The `AppModeBean`, which displays Edit mode

This section describes how to create these objects. It contains the following subsections:

- Section 23.3.7.1, "How to Create the AppNavigationUtils Class"
- Section 23.3.7.2, "How to Create AppCloseHandler"
- Section 23.3.7.3, "How to Register the AppCloseHandler"
- Section 23.3.7.4, "How to Create AppModeBean"

23.3.7.1 How to Create the AppNavigationUtils Class

To create the `AppNavigationUtils` class:

1. From the File menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.
3. In the **Name** field, enter `AppNavigationUtils` and click **OK**.

   The `AppNavigationUtils.java` file is rendered in the Source view.
4. Import the following libraries:

```
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpServletRequest;
```

5. Add the following code:

```
public class AppNavigationUtils
{

  public static void redirectToSamePage()
  {
    HttpServletRequest request =
(HttpServletRequest)FacesContext.getCurrentInstance().getExternalContext().getR
equest();
    String url = request.getRequestURL().toString();
    String _adfCtrlState = request.getParameter("_adf.ctrl-state");
    url = url + "?_adf.ctrl-state="+ _adfCtrlState;
    System.out.println(url);
    try
    {
      FacesContext.getCurrentInstance().getExternalContext().redirect(url);
      FacesContext.getCurrentInstance().responseComplete();
    }
    catch(Exception e)
    {
      e.printStackTrace();
    }
  }
}
```

23.3.7.2 How to Create AppCloseHandler

To create `AppCloseHandler`:

1. From the File menu, choose **New**.

2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.

3. In the **Name** field, enter `AppCloseHandler` and click **OK**.

   The `AppCloseHandler.java` file displays in the Source view.

4. Import the following libraries:

```
import oracle.adf.view.page.editor.event.CloseEvent;
import oracle.adf.view.page.editor.event.CloseListener;
```

5. Add the following code:

```
public class AppCloseHandler
  implements CloseListener
{

  public void processClose(CloseEvent closeEvent)
  {
    AppNavigationUtils.redirectToSamePage();
  }
}
```

23.3.7.3 How to Register the AppCloseHandler

After creating `AppCloseHandler`, you must register it in the Composer extension file, `pe_ext.xml`.

To register the event handler:

1. In the `pe_ext.xml` file, add the following entries:

```
<event-handlers>
  <event-handler event="close">view.AppCloseHandler</event-handler>
</event-handlers>
```

2. Save the file.

For more information about event handlers, see Section 21.6, "Configuring Event Handlers for Composer UI Events."

23.3.7.4 How to Create AppModeBean

To create `AppModeBean`:

1. From the **File** menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.

3. In the **Name** field, enter `AppModeBean` and click **OK**.

   The `AppModeBean.java` file displays in the Source view.

4. Import the following libraries:

```
import javax.faces.event.ActionEvent;
import oracle.adf.view.page.editor.mode.ModeContext;
```

5. Add the following code:

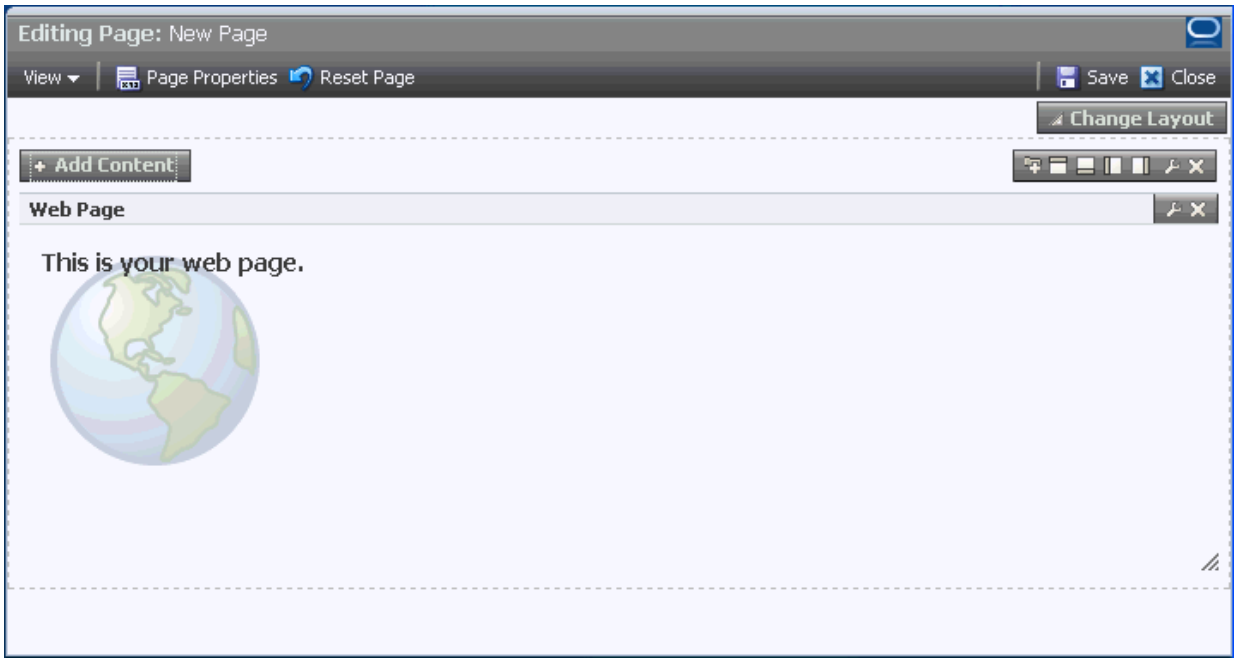
```
public class AppModeBean
{

  public void edit(ActionEvent actionEvent)
  {
    ModeContext.getCurrent().setEditMode();
    AppNavigationUtils.redirectToSamePage();
  }
}
```

23.3.8 What Happens at Runtime

To see how the `site` customization layer functions, first run the JSF page in a browser. Then log in to the page as `admin`, and click the **Edit** link to switch to Edit mode. At runtime, customize the page. For example, drop a Web page, movable box, or an image onto the page. The page should look like Figure 23–4.

Figure 23–4 Customized Page



Go to `/mds/mdssys/cust/site/webcenter` in your application directory, and open the *`pagename`*`.jspx.xml` file. This is where the site-level customizations (application customizations) that you made to the page are stored.

To use the `user` customization layer, log in to the page as `scott` and personalize the page in View mode. Then go to `/mds/mdssys/cust/user/scott` in your application directory, and open the *`pagename`*`.jspx.xml` file. This is where the user-level customizations that you made to the page are stored.

23.4 Troubleshooting MDS-Related Problems with Composer

This section provides information to assist you in troubleshooting MDS-related problems you may encounter while using Composer.

For information about configuring logging, see "Configuring ADF Logging for Composer".

Problem

Your application is configured to use MDS sandbox. When you run the application, the sandbox either does not work or generates exceptions.

Solution

Ensure the following:

- A database repository is used. Sandbox works only with a database-based repository, and not with a file-based repository.
- Ensure that the order of filters in `web.xml` is correct. The `<filter>` entries must be in the correct order for sandbox to work correctly.

For information, see Section 23.2, "Using Composer Sandbox."

24

Modifying Default Security Behavior of Composer Components

You can override the default security definitions at various levels in keeping with your business requirement. For information about the default security behavior of Composer components, see Section 18.7, "Security and Composer."

This chapter describes how to override the default security behavior of Composer components. It contains the following subsections:

- Section 24.1, "Applying Component-Level Restrictions by Defining Customization Policies"
- Section 24.2, "Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example"
- Section 24.3, "Applying Tag-Level Security Using the customizationAllowed Attribute"
- Section 24.4, "Applying Attribute-Level Security"
- Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions"
- Section 24.6, "Implementing Task Flow Security"
- Section 24.7, "Overriding Composer's Default Security Policies"

24.1 Applying Component-Level Restrictions by Defining Customization Policies

By default, the MDS restricts application customization of page components, that is, users cannot customize components or their attributes at runtime. To enable application customization, you must lift the default restrictions on components and their attributes. When you add a `Page Customizable` component to the page and populate it with content, the default customization restrictions on the `Page Customizable` and all its child components are lifted. However, in some instances, such as when the `Page Customizable` component is part of the template used for the page, you may need to explicitly allow application customization of components and their attributes. This section describes how to apply MDS type-level restrictions or instance-level restrictions on components and their attributes. It contains the following subsections:

- Section 24.1.1, "How to Define Type-Level Customization Policies"
- Section 24.1.2, "How to Define Instance-Level Customization Policies"

Note: For information about creating customizable pages using page templates, see Section 20.1.7, "How to Create a Page Template for Creating Customizable Pages."

24.1.1 How to Define Type-Level Customization Policies

When you apply type-level restrictions on a component, all instances of the component are restricted. This is useful if you want to allow only a specific set of users to edit a particular component or its attributes while restricting all other users from editing it. For example, if you want to allow only a page creator or application administrator to change the page layout at runtime, you can define type-level restrictions on the `Layout Customizable` component and enable only users with `admin` role to edit this component.

You can enable type-level restrictions on components and their attributes in a standalone XML file and then register this file in the `mds-config` section of the `adf-config.xml` file or in an `mds-config.xml` file. The standalone XML file contains annotations that match the types for which customization restrictions must be specified.

You can create an `mds-config.xml` file in your application's `META-INF` directory and register the standalone file in the `mds-config.xml` file.

To enable application customization of selected components or attributes at the type level:

1. Create an XML document, for example `standalone.xml`, add the following code, and replace the text in bold with appropriate values:

```
<?xml version="1.0" encoding="UTF-8" ?>
<grammarMetadata xmlns="http://xmlns.oracle.com/bali/xml/metadata"
                 xmlns:mds="http://xmlns.oracle.com/mds"
                 namespace="tag_namespace">
  <elementMetadata elementName="component_name">
    <mds:customizationAllowed>true</mds:customizationAllowed>
      <attributeMetadata attributeName="attribute_name">
        <mds:customizationAllowedBy>security_roles</mds:customizationAllowedBy>
      </attributeMetadata>
  </elementMetadata>
</grammarMetadata>
```

   The `<customizationAllowedBy>` tag can appear only once. Multiple values can be specified as a space-separated list of allowed policies as part of the declaration. To enable or disable application customization for specific users or roles, the users or roles must be included in the customization policy of the MDS session. For more information, see Section 24.2.3, "How to Customize the SessionOptions Object to Include Customization Policy."

   The `<customizationAllowed>` tag takes a Boolean as its value. A value of `true` for this tag means that anyone can customize the specified component, so long as other inherited customization polices allow it.

   The `<customizationAllowedBy>` tags do not serve any purpose if the same object has been tagged with `<customizationAllowed>false</customizationAllowed>`. Additionally, the customization must be allowed at the top level of the object tree. Otherwise the default behavior dictates that no customization can be permitted at a lower level.

2. Register this XML file in either of the following ways:

To register the XML file in the `adf-config.xml` file, add the `type-config` element inside the `mds-config` section using the following format:

```
<mds-config xmlns="http://xmlns.oracle.com/mds/config">
  <type-config>
    <standalone-definitions>
      <classpath>File_Path/standalone.xml</classpath>
    </standalone-definitions>
  </type-config>
</mds-config>
```

Create an `mds-config.xml` file in your application's `META-INF` folder and add a `type-config` element inside the `mds-component-config` section of the file in the following format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<mds-component-config version="11.1.1.000"
xmlns="http://xmlns.oracle.com/mds/config">
  <type-config>
    <standalone-definitions>
      <classpath>File_Path/standalone.xml</classpath>
    </standalone-definitions>
  </type-config>
</mds-component-config>
```

3. Make the standalone files and associated XSDs available in a shared library JAR file; otherwise MDS cannot load them.

   At runtime, MDS searches all `mds-config` instances in the `META-INF` directories and loads all the customization restrictions specified in standalone files.

24.1.2 How to Define Instance-Level Customization Policies

Instance-level restrictions are useful if you want to allow or restrict customization of a particular instance of a component. The code for instance-level customization policies is similar to that used in the type-level policy definitions. However, instance level policies override type-level policies. That is, instance-level restrictions hold true irrespective of the restrictions from the type. For example, if you have used a form with some UI elements in your login page and in other pages such as a user preferences page, you may want to allow users to customize all instances of the form except on the login page. For this, you can allow customization of the form at the type level and restrict customization only on the instance in the login page.

You can enable customization on a component or any of its attributes at the instance level. To apply customization restriction at the component level, you can set the `customizationAllowed` and `customizationAllowedBy` attributes on the component in JDeveloper (see Section 24.1.1, "How to Define Type-Level Customization Policies"). To apply restrictions on a component's attributes, perform the steps described in this section.

Instance-level policies for component attributes are defined in an RDF file. The RDF file is picked up by MDS, and the policies are implemented.

To define instance-level customization policies:

1. Create an RDF file with the same name as the JSPX file, but using the RDF extension, for example `main.jspx.rdf`.

   Create this file in the *`application_home`*`/project/public_html/mdssys/mdx` folder. Creating the RDF file in this folder structure ensures that the file is picked up by MDS and the policies defined here are implemented.

2. Include tags in the following format to apply restrictions on a component's attribute:

   tag_id(xmlns(*tag_prefix*=*tag_namespace*))/@*tag_prefix*:*attribute_name*

   The following example shows how to apply a restriction on a `Go Image Link` component and the `text` attribute of a `Command Button` component:

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="/"/>
<!-- Opens the customization on a tag with ID 'gil1'.  -->
<rdf:Description rdf:about="gil1">
  <customizationAllowed
xmlns="http://xmlns.oracle.com/mds">false</customizationAllowed>
</rdf:Description>

<!-- Restricts customizations on a commandButton component's 'text' attribute.
-->
<rdf:Description rdf:about="cb1/@text">
  <customizationAllowed
xmlns="http://xmlns.oracle.com/mds">false</customizationAllowed>
</rdf:Description>
```

At runtime, these policies are read and the artifacts are enabled for customization accordingly.

24.2 Applying a Component Instance-Level Customization Restriction By Using Security Roles: Example

This section explains how you can use component-level restrictions to limit access to certain component functions, based on different user roles and responsibilities. Specifically, it describes how to apply an instance-level customization restriction to a `Panel Customizable` component to provide customization access to only users with `admin` role. That is, when a user with `admin` role logs into the system at runtime, the user can customize the `Panel Customizable` on which the instance-level customization is applied.

In Composer's Property Inspector, the properties of the `Panel Customizable` are not displayed to users who do not have `admin` privileges. The Edit icon or menu item is disabled on restricted components. These users cannot drop any `Movable Box` into the `Panel Customizable` because of the applied restriction. Moreover, they cannot rearrange the `Show Detail Frames` inside this `Panel Customizable`.

To restrict a functionality based on user roles is one way to apply customization restrictions. You can also use expression language expressions (ELs), which provide roles-based results, to enable or restrict access to a component function.

> **Note:** MDS customization restrictions are natively honored only by `Panel Customizable`, `Show Detail Frame`, and `Layout Customizable` components, and not ADF Faces components, such as `Splitter` and `showDetailItem`.

This section contains the following subsections:

- Section 24.2.1, "How to Configure ADF Security"

- Section 24.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File"
- Section 24.2.3, "How to Customize the SessionOptions Object to Include Customization Policy"
- Section 24.2.4, "How to Register the Implementation with Composer"
- Section 24.2.5, "How to Configure WebCenterComposerFilter"
- Section 24.2.6, "How to Apply an Instance-Level Customization Restriction"
- Section 24.2.7, "What Happens at Runtime"

24.2.1 How to Configure ADF Security

Before applying an instance-level customization restriction, you must secure your application using ADF security. For information, see Section 69.3, "Configuring ADF Security."

24.2.2 How to Define Roles and Grant Privileges in the jazn-data.xml File

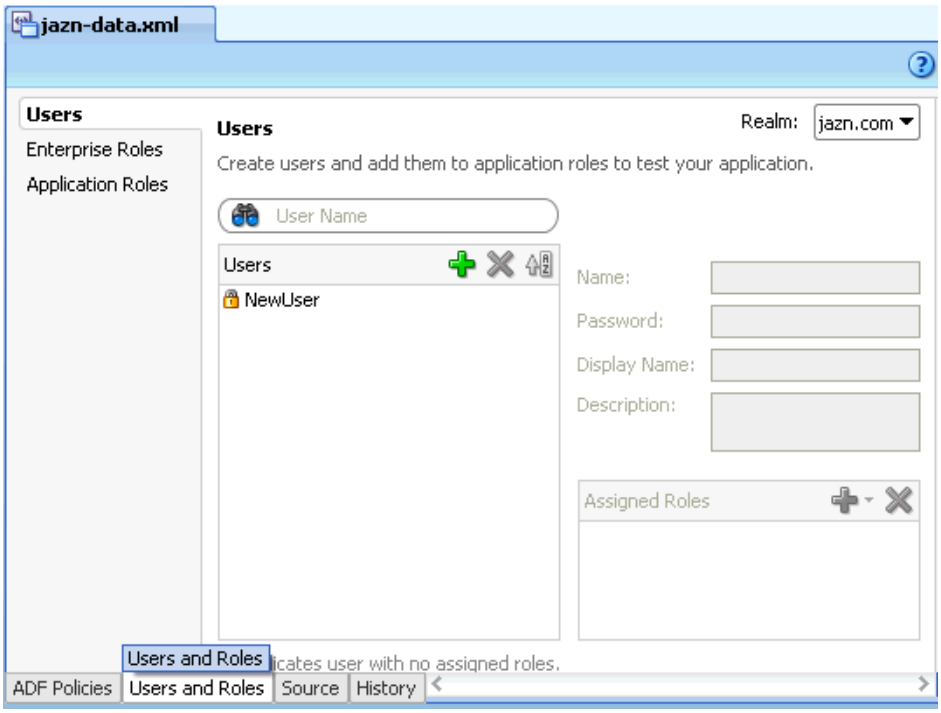
In a previous section, a customization restriction is applied to a `Panel Customizable` component so that, at runtime, only the user with `admin` role can access all the features of `Panel Customizable`, and therefore, can customize its attributes.

In this section, two roles, `admin` and `customer`, are created and page permissions are granted to both users. Users with `admin` role can customize the `Panel Customizable` (`panelCustomizable1`) component. However, because of the customization restriction you applied in an earlier section, users with `customer` role do not have the rights to customize `panelCustomizable1`, and these users cannot use all the features of `panelCustomizable1`.

To create roles in the `jazn-data.xml` file:

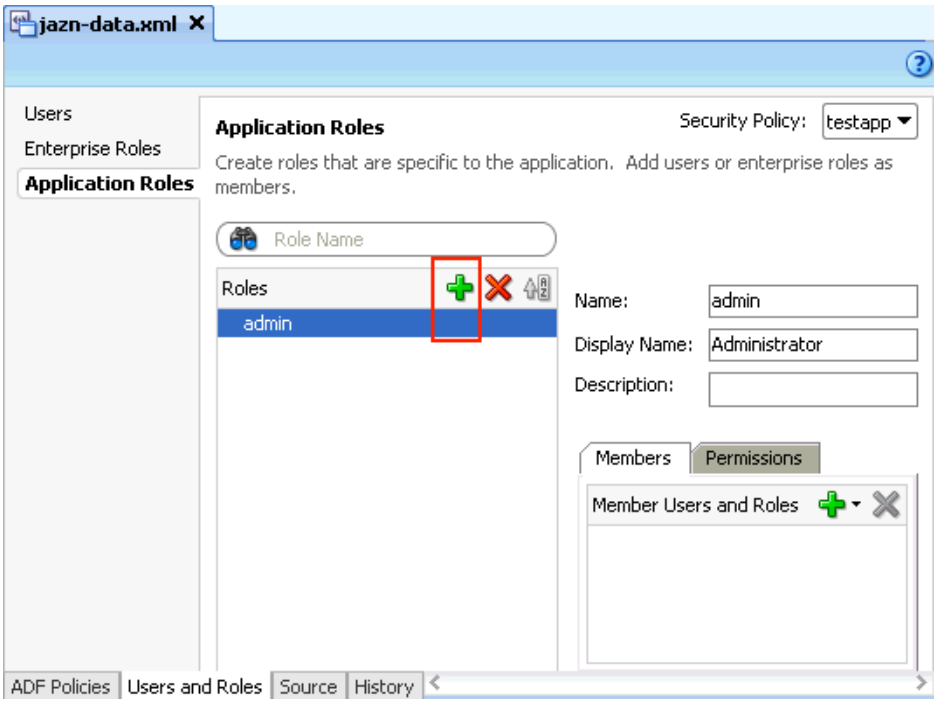
1. In the Application Resources panel, right-click the **jazn-data.xml** file and select **Open**.
2. Click the **Users and Roles** tab at the bottom of the editor to open the Users section of the file, shown in Figure 24–1.

Figure 24–1 The jazn-data.xml File



3. Click the **Application Roles** tab.
4. In the Application Roles page, click the **New Application Role** icon in the Roles table, shown in Figure 24–2.

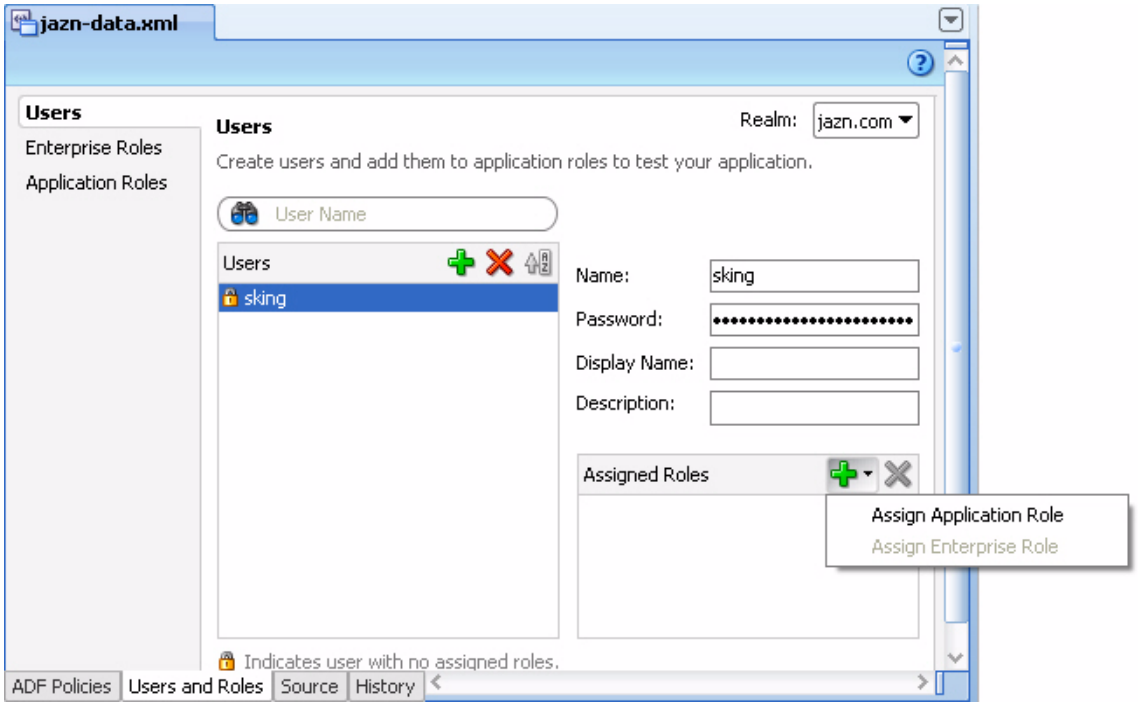
Figure 24–2 Roles Section in the jazn-data.xml File



5. In the **Name** field, specify the user name, `admin`.

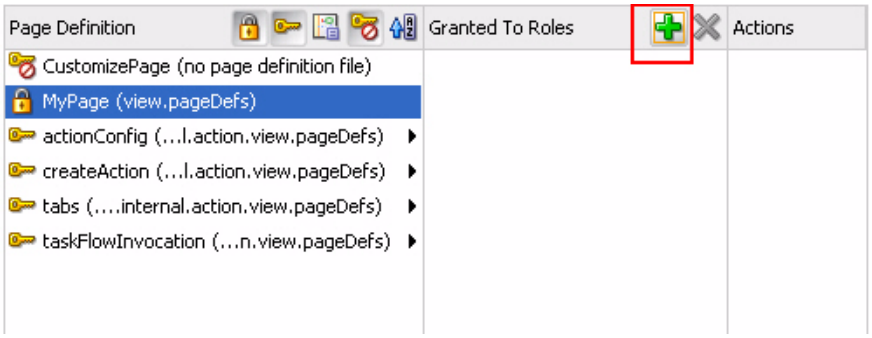
6. Optionally, in the Display Name and Description fields, specify a display name and description for the new role.
7. Repeat steps 3 through 6 to add another role, `customer`.
8. Click the **Users** tab.
9. In the Users page, create new users by clicking the **New User** icon and specifying credentials, and assign roles to these users by clicking the **Assign Roles** icon in the Assigned Roles table and selecting **Assign Application Role**, as shown in Figure 24–3.

Figure 24–3 Users Section in the jazn-data.xml File



10. Click the **ADF Policies** tab at the bottom of the page.
11. Select the **Web Pages** tab on the page.
12. From the list of page definitions, select the page on which you want to grant privileges, as shown in Figure 24–4.

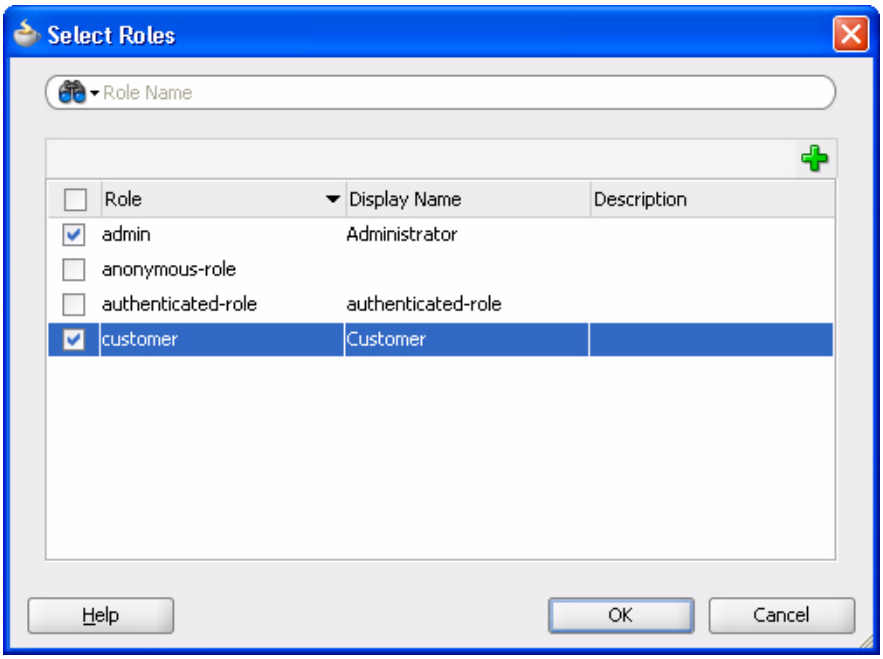
Figure 24–4 Page Definitions in the jazn-data.xml File



13. In the Granted to Roles column, click the **Add Application Role** icon.

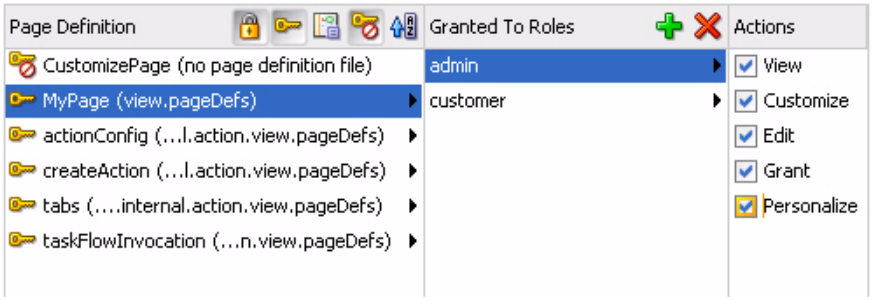
14. In the Select Roles dialog (Figure 24–5), select the `admin` and `customer` roles and click **OK**.

Figure 24–5 Select Roles Dialog



15. Select each role and grant all the permissions (View, Customize, Edit, Grant, Personalize), as shown in Figure 24–6.

Figure 24–6 Permissions List in the jazn-data.xml File



For more information about creating users and roles, see the section *"Enabling ADF Security in a Fusion Web Application"* in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

24.2.3 How to Customize the SessionOptions Object to Include Customization Policy

To enable the expected runtime behavior of the `Panel Customizable` component, where only the user with `admin` role has access to all capabilities of the `Panel Customizable`, including the ability to customize it, the MDS session for this request must include the user roles in its customization policy. To achieve this, the following customization policy is included in the `SessionOptions` object:

```
SecurityContext stx = ADFContext.getCurrent().getSecurityContext();
  cPol = new CustomizationPolicy(stx.getUserRoles());
```

Then the MDS session is created with this customization policy.

To implement the `ComposerSessionOptionsFactory` class:

1. From the File menu, choose **New**.
2. In the New Gallery dialog, expand **General**, select **Java**, then **Java Class**, and click **OK**.

   The Create Java Class dialog opens.
3. In the **Name** field, enter `AppsSessionOptionsFactoryImpl`.
4. Click **OK.**

   The `AppsSessionOptionsFactoryImpl.java` file is rendered in the Source view.
5. Import the following libraries:

```
import oracle.adf.share.ADFContext;
import oracle.adf.share.security.SecurityContext;
import oracle.adf.view.page.editor.mds.ComposerSessionOptionsFactory;
import oracle.adf.view.page.editor.mode.ModeContext;
import oracle.mds.config.CustClassListMapping;
import oracle.mds.config.CustConfig;
import oracle.mds.core.SessionOptions;
import oracle.mds.cust.CustClassList;
import oracle.mds.cust.CustomizationClass;
import oracle.mds.cust.CustomizationPolicy;
```

6. Add the following code to implement the `ComposerSessionOptionsFactory` class and provide `SessionOptions`:

```
public class AppsSessionOptionsFactoryImpl
implements ComposerSessionOptionsFactory
{
  public AppsSessionOptionsFactoryImpl()
  {
  }
  public SessionOptions createSessionOptions(SessionOptions sessionOptions,
                                             String mode)
  {
    CustomizationClass[] custLayer;
    CustConfig custConfig = null;
    CustomizationPolicy cPol = null;

    if (ModeContext.EDIT_MODE.equals(mode))
    {
      //Mode is Edit, change to SiteCC
      custLayer = EDIT_LAYER;
    }
    else
    {
      //Mode is View, change to UserCC + SiteCC
      custLayer = VIEW_LAYER;
    }
    try
    {
      CustClassList custClassList = new CustClassList(custLayer);
      CustClassListMapping custClassListMapping =
        new CustClassListMapping("/", null, null, custClassList);
      custConfig = new CustConfig(new CustClassListMapping[]
```

```
            { custClassListMapping });
      SecurityContext stx = ADFContext.getCurrent().getSecurityContext();
      cPol = new CustomizationPolicy(stx.getUserRoles());
    }
    catch (Exception e)
    {
      e.printStackTrace();
    }
    if(sessionOptions.getServletContextAsObject() != null)
    {
      return new SessionOptions(sessionOptions.getIsolationLevel(),
                                sessionOptions.getLocale(), custConfig,
                                sessionOptions.getVersionContext(),
                                sessionOptions.getVersionCreatorName(),
                                cPol == null ?
sessionOptions.getCustomizationPolicy() : cPol,
                                sessionOptions.getServletContextAsObject());
    }
    else
    {
      return new SessionOptions(sessionOptions.getIsolationLevel(),
                                sessionOptions.getLocale(), custConfig,
                                sessionOptions.getVersionContext(),
                                sessionOptions.getVersionCreatorName(),
                                cPol == null ?
sessionOptions.getCustomizationPolicy() : cPol);
    }
  }
  //Edit mode SiteCC
  private static final CustomizationClass[] EDIT_LAYER =
    new CustomizationClass[]
    { new SiteCC() };
  //View mode SiteCC + USerCC
  private static final CustomizationClass[] VIEW_LAYER =
    new CustomizationClass[]
    { new SiteCC(), new UserCC() };
}
```

24.2.4 How to Register the Implementation with Composer

Perform the steps described in Section 23.3.5, "How to Register the Implementation with Composer" to register the `ComposerSessionOptionsFactory` class implementation, where user roles were included in the MDS session's customization policy.

24.2.5 How to Configure WebCenterComposerFilter

To configure `WebCenterComposerFilter`, perform the steps described in Section 23.3.6, "How to Configure WebCenterComposerFilter."

24.2.6 How to Apply an Instance-Level Customization Restriction

In this section, you enable application customization for particular role on a `Panel Customizable` component. At runtime, only the specified role, for example `admin`, can access all the features of the `Panel Customizable`, including customizing its attributes. Users with any other roles can use only a limited set of component features.

To apply an instance-level customization restriction:

1. In the JSPX page, add a **Change Mode Link** component from the Composer tag library.
2. Add a **Page Customizable** component below the `Change Mode Link` component.
3. Apply an instance-level customization restriction on the component `Panel Customizable` in the Page Customizable.

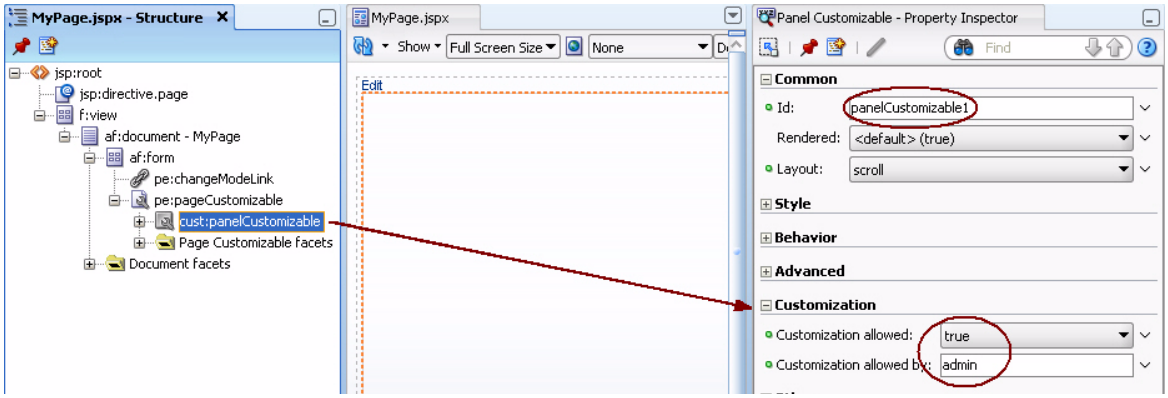
   This restriction is stored in the following directory: `public_html/mdssys/mdx/<pagename>.jspx.rdf`.

   The Source view should look like this:

```
<pe:changeModeLink id="cml1"/>
<pe:pageCustomizable id="pageCustomizable1">
  <cust:panelCustomizable id="panelCustomizable1" layout="scroll"/>
    <f:facet name="editor">
      <pe:pageEditorPanel id="pep1"/>
    </f:facet>
</pe:pageCustomizable>
```

   The design view should look like Figure 24–7.

Figure 24–7 Design View: Customization Allowed



4. Add a `Show Detail Frame` component within the `Panel Customizable` and include another `Panel Customizable` component as a child of the `Show Detail Frame` component.

> **Note:** This step is for illustration purpose only as it is easier to explain runtime behavior by showing multiple components. For detailed information on how to add Composer components, see Section 5.3, "Adding Pages to a Portal."

24.2.7 What Happens at Runtime

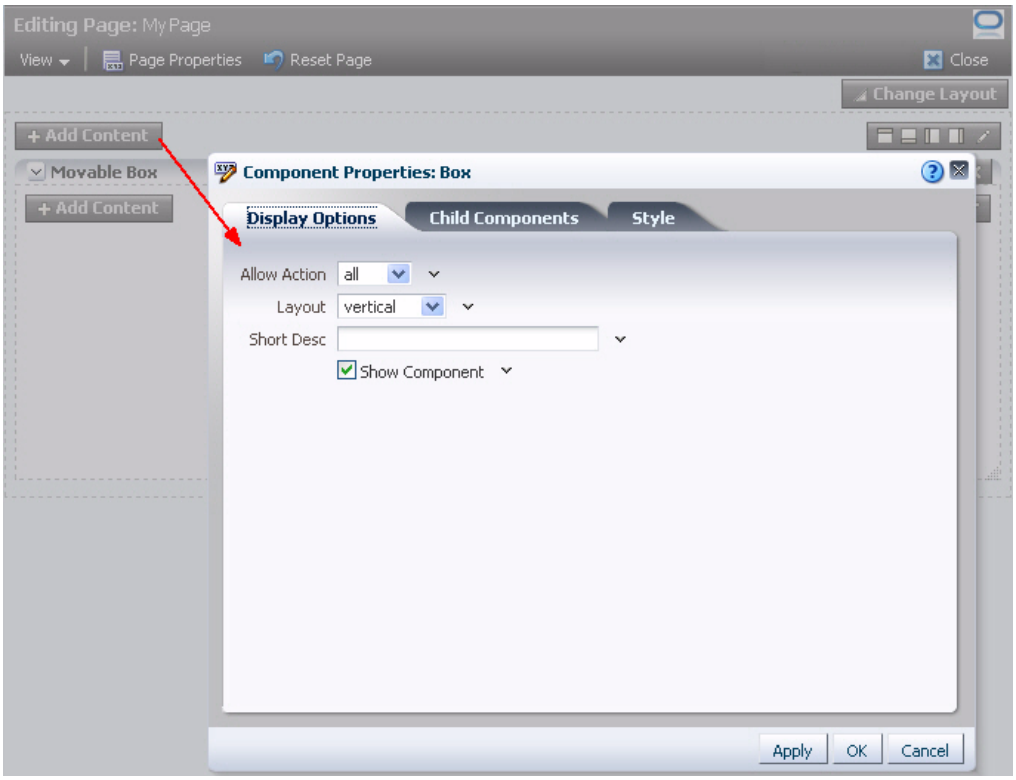
What happens at runtime is influenced by which user is logged on. To see the difference, log in as one user and then the other. For example:

In the Application Navigator, right-click the JSPX page and select **Run**.

Log in to the page as a user with `admin` role, and switch to Edit mode. The **Add Content** button and the **Edit** icon are displayed on the outer and inner `Panel Customizable` components. As a user with `admin` role, you can switch to Source

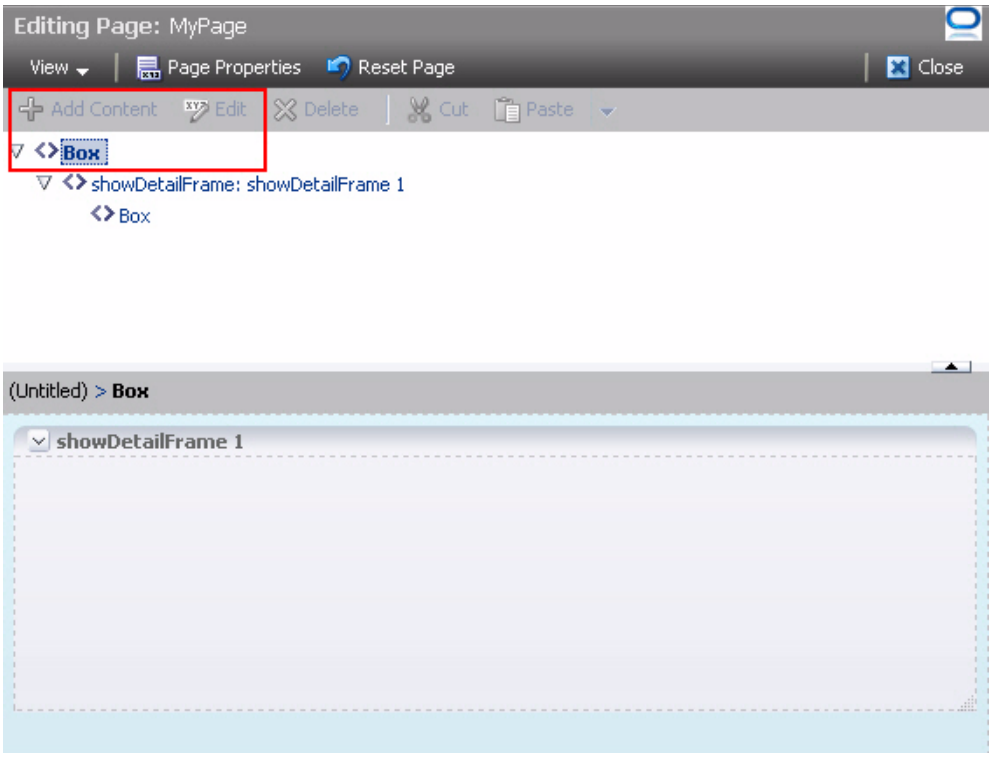
view and access the properties of the outer `Panel Customizable`, as shown in Figure 24–8. That is, users with `admin` role can edit this component.

Figure 24–8 Edit Mode for admin



Log in to the page as a user with `customer` role, and switch to Edit mode. The **Add Content** button and the **Edit** icon are not available on the outer `Panel Customizable`. That is, this component is not editable for users with `customer` role. In Source view, the Add Content and Edit icons are grayed out for the component, as shown in Figure 24–9.

Figure 24–9 Restricted Component in Page Source View



24.3 Applying Tag-Level Security Using the customizationAllowed Attribute

By default, a `Page Customizable` component enables application customization on all components under it. You may want to change this to restrict customization on some components. The `Page Customizable` component does not enable customization on components that are on a nested page or fragment. You must enable customization on such components manually. To address such requirements, MDS provides the `customizationAllowed` and `customizationAllowedBy` attributes. These attributes can be used to enable or restrict customization on specific component instances on a page.

The `customizationAllowed` attribute controls whether the component can be customized at runtime. If you set this attribute to `false`, then the component cannot be customized at runtime. That is, in the Component Properties dialog the properties are grayed out and do not allow editing.

The `customizationAllowedBy` attribute specifies the roles for which customization is enabled.

This section provides examples of enabling and restricting customization on a component. It contains the following subsections:

- Section 24.3.1, "How to Enable Application Customization on an Image Component"
- Section 24.3.2, "How to Restrict Customization on an Image Component"

For more information, see "Extended Metadata Properties" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

24.3.1 How to Enable Application Customization on an Image Component

To enable application customization on an Image component:

1. In the `MyPage.jspx` page that you created in Section 20.2, "Designing Editable Pages Using Composer Components: Example," add an ADF Faces Image component, for example, `brandingImage.gif`, inside the `Panel Customizable` called `panelCustomizable1`.

   Application customization is enabled automatically on the `Image` component since it is nested inside a `Page Customizable` component.

2. Run `MyPage.jspx`.
3. Switch to Edit mode of the page.
4. From the **View** menu, select **Source** to switch to Source view of the page.
5. Select the image in the component hierarchy in the Source View panel and click the **Show the properties of Image** icon.

   The Component Properties dialog displays the image properties. You can edit any available property in this dialog. Edit a property and click **OK**. The property's editable value demonstrates that application customization is enabled on the component.

24.3.2 How to Restrict Customization on an Image Component

You can restrict customization of an image on the page by setting the `customizationAllowed` attribute to `false` on the image component.

To restrict customization on the Image Component:

1. In Oracle JDeveloper, select the `Image` component that you added in the previous section, and in the Property Inspector set the value for **customizationAllowed** to `false`.
2. Run `MyPage.jspx`.
3. Switch to page Edit mode.
4. From the **View** menu, select **Source** to switch to the Source view of the page.
5. Select the image in the component hierarchy in the Source View panel, and click the **Show the properties of Image** icon.

   In the Component Properties dialog, the properties are grayed out and cannot be edited.

24.4 Applying Attribute-Level Security

You can choose whether specific component attributes can be customized at runtime. Additionally, you can specify whether the changes made to an attribute must be persisted to a persistence store, such as MDS. You can apply attribute-level security in the following ways:

- By defining property filters
- By persisting changes at the component level

For information about applying property filters, see Section 21.8.1, "How to Define Property Filters."

This section describes how to persist changes at the component level. It contains the following subsections:

- Section 24.4.1, "How to Define Change Persistence at the Component Level"
- Section 24.4.2, "What Happens at Runtime"

24.4.1 How to Define Change Persistence at the Component Level

If your application is configured to use `FilteredPersistenceChangeManager` to persist changes, you can use the `persist-changes` attribute in the `adf-config.xml` file to specify whether changes to a specific attribute must be persisted.

> **Note:** For information about change persistence, see Section 22.11, "Configuring the Persistence Change Manager."
>
> For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

Example 24–1 shows a sample `adf-config.xml` file with `persist-changes` attributes defined for some attributes.

Example 24–1 Using the persist-changes Attribute in the adf-config.xml File

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config">
  <adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
    <persistent-change-manager>
      <persistent-change-manager-class>
       oracle.adf.view.rich.change.MDSDocumentChangeManager
      </persistent-change-manager-class>
    </persistent-change-manager>
    <taglib-config>
      <taglib uri="http://xmlns.oracle.com/adf/pageeditor">
        <tag name="imageLink">
          <attribute name="destination">
            <persist-changes>true</persist-changes>
          </attribute>
        </tag>
        <tag name="customAction">
          <attribute name="text">
            <persist-changes>true</persist-changes>
          </attribute>
          <attribute name="icon">
            <persist-changes>true</persist-changes>
          </attribute>
        </tag>
      </taglib>
    </taglib-config>
  </adf-faces-config>
</adf-config>
```

24.4.2 What Happens at Runtime

If you set `persist-changes` to `true`, then a change made to the component attribute is persisted in any instance of that component. If you set `persist-changes` to `false`, then even if you change the attribute value in the Component Properties dialog at runtime, the change is not saved.

24.5 Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions

The ability of a user to perform actions on `Panel Customizable` and `Show Detail Frame` components is inherited from page security. Inheritance is based on the value of the application-wide switch, `enableSecurity`, in the `adf-config.xml` file. If you select the WebCenter Portal - Framework Application template when you create your application, then the `adf-config.xml` file is located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.

This section describes how to apply action-level restrictions on `Panel Customizable` and `Show Detail Frame` component actions. It contains the following subsections:

- Section 24.5.1, "How to Add an enableSecurity Section to adf-config.xml"
- Section 24.5.2, "Defining Security at the Actions Category Level"
- Section 24.5.3, "Defining Security at the Actions Level"

24.5.1 How to Add an enableSecurity Section to adf-config.xml

The `enableSecurity` element is not available by default in `adf-config.xml`. To override or extend the page-level security inheritance for `Panel Customizable` and `Show Detail Frame` components, you must add the `customizableComponentsSecurity` section in the `adf-config.xml` file, as shown in Example 24–2, and set the `enableSecurity` element in that section to `true`.

Example 24–2 enableSecurity Element in the Customizable Components Security Section in adf-config.xml

```
<cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <cust:enableSecurity value="true"/>
    <cust:actionsCategory>
       ......................................
</cust:customizableComponentsSecurity>
```

Restrictions on actions can be implemented at the following levels:

- **Page level**: You can define security for WebCenter Portal Customizable Components so that page-level privileges are inherited by these components. This is the default behavior.

  By default, customizable components inherit allowable actions from the defined page-level permissions, such as edit, personalize, or customize. That is, a user who has edit, personalize, or customize privileges on a page can perform View mode user customizations on that page. The `enableSecurity` element enables you to override the security inheritance behavior. It can take either of the following values:

  - `true`: If set to `true` (the default when not specified), then the ability of a user to modify a component is first determined from the page permissions and then adjusted according to the current set of actions defined for that type of permission. For example, if a user has customize permission, then the actions that constitute the customize category (move, customize, and so on) are available to the user, but they are overridden by the actions that are defined in the `adf-config.xml` file.

  - `false`: If set to `false`, then all actions are available to users. A user's page permissions and actions configured in `adf-config.xml` are ignored.

- **Actions category level**: You can define security on all actions for `Panel Customizable` and `Show Detail Frame` components.

  You can add an `actionsCategory` element in the `adf-config.xml` file to define security on multiple actions simultaneously. Depending on the `actionCategory` attributes that you enable, appropriate privileges are provided on the components.

- **Actions level**: You can define security on individual actions for `Panel Customizable` and `Show Detail Frame` components.

  You can use the `actions` element in the `adf-config.xml` file to enable or disable individual actions. Depending on the `actions` attributes that you enable, appropriate privileges are provided on the components.

Notes:

- If you set `enableSecurity` to `true` in the `adf-config.xml` file and define restrictions on actions or action categories, then for these restrictions to take effect you must grant Personalize or Customize privilege on the page, in addition to View privilege.

  If you are not using the `enableSecurity` element or it is set to `false`, then page-level permissions are honored. In such a case, having View privilege on the page is sufficient as all customizable components actions are rendered by default on the page.

- Privileges can be inherited from the parent only. Inheritance from a component in any other position in the hierarchy is not supported.

- Restrictions defined at the actions category level or actions level are applicable to all instances of the `Panel Customizable` or `Show Detail Frame` component across the application. You cannot apply restrictions on a single instance of the component.

You can define security for component actions at the application level if `enableSecurity` is set to `true` in the `adf-config.xml` file. A value of `true` implies that permission checks are made in addition to the `actionsCategory` and `actions` values specified in the `adf-config.xml`.

24.5.2 Defining Security at the Actions Category Level

You can add an `actionsCategory` element in the customizable components actions security section in the `adf-config.xml` file to define security on multiple `Panel Customizable` and `Show Detail Frame` components actions. Depending on the `actionsCategory` attributes that you enable, appropriate privileges are provided on the components.

The `actions` and `actionsCategory` elements in the `adf-config.xml` file have certain default mappings. Table 24–1 describes the different `actionsCategory` attributes and the actions they support by default.

For information about the Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

Table 24–1 Actions Categories and Actions Mapping

actionsCategory	Actions Supported
`personalizeActionsCategory`	showMoveAction showRemoveAction showMinimizeAction showResizer allowAction
`customizeActionsCategory`	showEditAction showAddContentAction showSplitAction

The behavior of components based on a combination of `personalizeActionsCategory` and `customizeActionsCategory` settings is as follows:

- If both `personalizeActionsCategory` and `customizeActionsCategory` are set to `false`, then users cannot move, expand, collapse, delete, resize, or edit `Show Detail Frame` components and they also cannot edit or delete `Panel Customizable` components or add content inside them.
- If `personalizeActionsCategory` is set to `true` and `customizeActionsCategory` is set to `false`, then users can move, expand, collapse, delete, or resize `Show Detail Frame` components but not edit them. Additionally, they can delete `Panel Customizable` components and rearrange components inside them but not edit them or add components to the page.

  Note: The `allowAction` setting, along with other actions settings, controls the ability to add content to the page and move or rearrange content on the page. For more information, see Section 24.5.3, "Defining Security at the Actions Level."

- If both `personalizeActionsCategory` and `customizeActionsCategory` are set to `true`, then users can move, expand, collapse, delete, resize, and edit `Show Detail Frame` components and they can also edit and delete `Panel Customizable` components and add content inside them. Users can add content from the Resource Catalog using the **Add Content** button, add Box components using the Add Box icons, and move components from other `Panel Customizable` components by dragging and dropping them.

  Note: It is not recommended that you set `personalizeActionsCategory` to `false` and `customizeActionsCategory` to `true` as privileges cascade from Grant to Customize to Personalize to View. That is, when you enable Customization, the user must already have Personalize privilege on the page.

Example 24–3 shows the `actionsCategory` entry that you can add to the customizable components actions security section in the `adf-config.xml` file.

Example 24–3 actionsCategory Element in the Customizable Components Security Section

```
<cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <cust:enableSecurity value="true"/>

  <cust:actionsCategory>
    <cust:actionCategory name="personalizeActionsCategory" value="true"/>
    <cust:actionCategory name="customizeActionsCategory" value="false"/>
  </cust:actionsCategory>

  <cust:actions>
  ......................................
  </cust:actions>

</cust:customizableComponentsSecurity>
```

You can also use EL for these element values, as shown in Example 24–4.

Example 24–4 EL Used in Customizable Components an actionCategory Entry

```
<cust:actionsCategory>
  <cust:actionCategory name="personalizeActionsCategory"
value="#{appBusinessRules.InsideCorpNetwork}"/>
</cust:actionsCategory>
```

For reference, the managed bean, `appBusinessRules`, is defined as shown in Example 69–7 in Section 69.15, "Overriding Inherited Security on Portlets and Customizable Components."

24.5.3 Defining Security at the Actions Level

You can use the `actions` element in the customizable components actions security section of the `adf-config.xml` file to show or hide individual `Show Detail Frame` actions. Depending on the `actions` attributes that you enable, appropriate privileges are provided on the `Show Detail Frame` components.

For information about Composer-specific configurations you can make in `adf-config.xml`, see Section B.2.2, "adf-config.xml."

The `allowAction` setting, along with other actions settings, controls the ability to add content to the page and move or rearrange page content.

In Composer, users can add and arrange content in the following ways:

- Add `Box` components using the **Add Box** icons.
- Add content inside a `Box` component, from the Resource Catalog, using the **Add Content** button.
- Move `Show Detail Frame` components within a `Panel Customizable` component by using the Move Up and Move Down actions on the `Show Detail Frame` component's Actions menu.
- Drag and drop `Show Detail Frame` components across `Panel Customizable` components.

Table 24–2 explains the ability to add content depending on the `allowAction` and `showAddContentAction` settings.

Table 24–2 Actions Settings for Adding Content to a Page

allowAction Setting	showAddContentAction Setting	Add Content Button Displayed?
`true`	`true`	Yes
`true`	`false`	No
`false`	true	No
`false`	`false`	No

Table 24–3 explains the ability to add content using the Add Box icons depending on the `allowAction` and `showSplitAction` settings.

Table 24–3 Actions Settings for Rearranging Content on a Page

allowAction Setting	showSplitAction Setting	Add Box Icons Displayed?
`true`	`true`	Yes
`true`	`false`	No
`false`	`true`	No
`false`	`false`	No

Table 24–4 explains the ability to move content on the page depending on the `allowAction` and `showMoveAction` settings.

Table 24–4 Actions Settings for Moving Content on a Page

allowAction Setting	showMoveAction Setting	Move Content on the Page?
`true`	`true`	Yes
`true`	`false`	No
`false`	`true`	No
`false`	`false`	No

Note:

- If you define security both at the actions category level and actions level and the values for some actions contradict, then a value of `false` takes precedence over `true`. For example, if you set `personalizeActionsCategory` to `true` and `allowAction` to `false`, then users cannot add content to their application pages at runtime. Similarly, if you set `personalizeActionsCategory` to `false` but set `allowAction` to `true`, then too users cannot add content to their application pages at runtime.
- Restrictions using actions settings are applicable only in Design view of a page. That is, if you set an action, such as `showEditAction` or `showRemoveAction`, on `Show Detail Frame` components to `false`, then users cannot edit or delete such components in Design view of the page. However, users can still select such components in Source view and edit or delete them.

Example 24–5 shows the `actions` entry that you can add to the customizable components actions section of the `adf-config.xml` file. You can use EL for element values.

Example 24–5 action Elements in the Customizable Components Security Section

```
<cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
  <cust:enableSecurity value="true"/>

  <cust:actionsCategory>
    ....................................
  </cust:actionsCategory>

  <cust:actions>
    <cust:action name="showMinimizeAction" value="true"/>
    <cust:action name="showMoveAction" value="false"/>
  </cust:actions>

</cust:customizableComponentsSecurity>
```

24.6 Implementing Task Flow Security

Since editing task flow permissions has a far reaching impact on the task flow editing experience, the Edit permission on task flows is disabled by default. In fact, no permissions are provisioned on task flows by default. To ensure that only users with appropriate permissions can edit a task flow, you can implement task flow security by performing the following two tasks in the Composer context:

- Explicitly grant Edit, Customize, or Grant permission on the task flow.
- Enable Composer to check for a task flow's permissions and allow only users with appropriate permissions to edit or customize the task flow.

This section explains how to apply restrictions and enable permission checks on task flows. It includes the following subsections:

- Section 24.6.1, "Granting Permissions on Task Flows"
- Section 24.6.2, "Enabling Task Flow Permission Check"

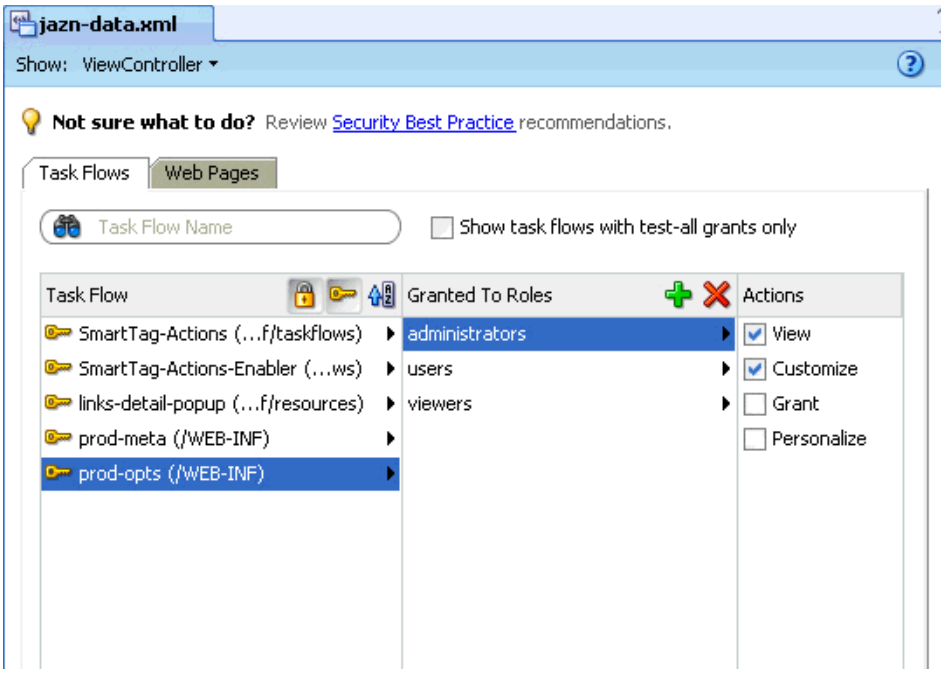
24.6.1 Granting Permissions on Task Flows

You can define the access policy for a task flow by creating permission grants in the Resource Grants page of the `jazn-data.xml` file overview editor. The grants you create will appear as metadata in the policy store section of the file. These grants, or permissions, control the ability to zoom into and edit the task flow in Composer.

To grant or revoke permissions on a task flow in the `jazn-data.xml` file:

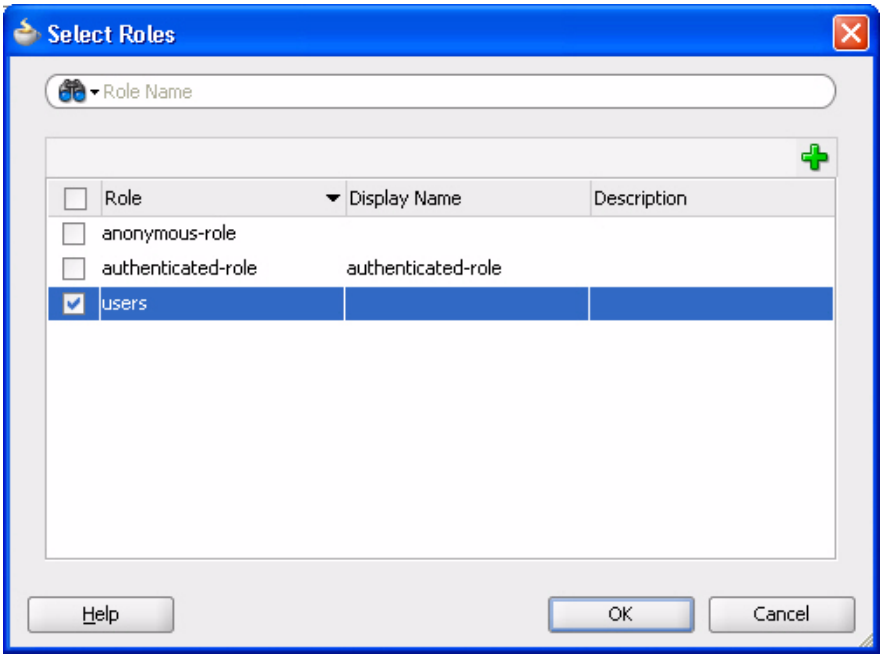
1. From the Application menu, select **Secure**, and then **ADF Policies**.
2. In the `jazn-data.xml` file, select the task flow in the Overview editor and click the **Add** button in the Granted to Roles column, shown in Figure 23–12.

Figure 24–10 Task Flow Security Region in the jazn-data.xml File



3. In the Select Roles dialog (Figure 23–13), select the application roles to which you want to grant permissions or create a new role by clicking the **Add** button.

Figure 24–11 Select Roles Dialog



4. Click **OK**.
5. Select one or more permissions in the Actions column.
6. Save the `jazn-data.xml` file.

Note: The permission grant would appear as follows in the source view of the file:

```
<permissions>
  <permission>

<class>oracle.adf.controller.security.TaskFlowPermission</class>

<name>/WEB-INF/task-flow-definition.xml#task-flow-definition</name>
    <actions>customize,view</actions>
  </permission>
</permissions>
```

The `TaskFlowPermission` class defines task flow-specific actions that it maps to the task flow's operations.

For more information about Composer's behavior based on task flow permissions, see Section 18.7.1, "Page and Task Flow Security."

For more information about defining permission grants on task flows, see the section titled "How to Define Security Policies for ADF Bounded Task Flows" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

24.6.2 Enabling Task Flow Permission Check

When a user tries to edit a task flow, Composer does not check the task flow's permissions by default. To enhance task flow security, you must configure Composer to check for a task flow's permissions and allow only users with the required privileges to edit it. You can do so by setting the `check-permission` property on the `<pe:task-flow-security>` element in the application's `adf-config.xml` file, as shown in the following example:

```
<pe:page-editor-config>
  <pe:security-config>
    <pe:task-flow-security check-permission="true" />
  </pe:security-config>
</pe:page-editor-config>
```

For more information about `adf-config.xml` settings, see Section B.2.2, "adf-config.xml."

Note: In earlier releases, you could enable permission checks on task flows by running the server with the following JVM parameter:

`-Doracle.composer.enableTaskflowPermissionCheck=true`

However, this method is now deprecated. For backward compatibility, Composer checks for the `enableTaskflowPermissionCheck` flag first, and then the setting in the `adf-config.xml` file.

24.7 Overriding Composer's Default Security Policies

Composer performs the following security checks to enable customization:

- Regions on the page have been granted edit or customize privilege.

- Task flows have been granted edit or customize privilege.

  Note: This is an optional security setting, which is not enabled by default.

- A component is not nested in an iterator.
- MDS customization restrictions have been applied on components and attributes.

Users can customize pages and task flows only if all the above criteria are met. By default, Composer uses `pagePolicy` to perform page-level security checks.

There may be situations where you want to perform different security checks; probably add a few application-level checks to what Composer already provides. You can do so by overriding Composer's default security policy with a custom policy. This section explains how to do this. It includes the following sections:

- Section 24.7.1, "How to Create a Custom Security Policy"
- Section 24.7.2, "How to Register a Custom Policy with Composer"
- Section 24.7.3, "What Happens at Runtime"

24.7.1 How to Create a Custom Security Policy

You can create custom policies by extending Composer's base security classes. Depending on whether you want to check pages or task flows, you can extend the following base security classes:

- Page Policy:
  `oracle.adf.view.page.editor.security.BasePageSecurityPolicy`
- Task Flow Policy:
  `oracle.adf.view.page.editor.security.TaskflowPermissionPolicy`

To take you one step further, you can configure your policy in such a way that Composer's default checks are performed in case the custom checks are not applicable or do not provide the desired results. For this, you must extend the `DefaultPageSecurityPolicy` class and call the `super.isCustomizable()` method, as shown in Example 24–6.

Example 24–6 Custom Policy Extending DefaultPageSecurityPolicy

```
public class CustomPageSecurityPolicy extends DefaultPageSecurityPolicy
{
  . . .

  public boolean isCustomizable()
  {
    // Security checks that this custom page policy must perform should go here
    . . .
    // If we cannot determine the security policy, chain it to Composer's default
implementation
    return super.isCustomizable();
  }
}
```

24.7.2 How to Register a Custom Policy with Composer

To configure a custom policy to override the Composer security policy, you must include the `<pe:security-config>` section in the application's `adf-config.xml` file and register the custom security policy.

To register a custom security policy:

1. Open the application's `adf-config.xml` file from the `ADF META-INF` folder under `Descriptors` in the Application Resources panel.
2. Add a `<pe:page-editor-config>` section if it does not already exist, and add the following code inside it:

```
<pe:security-config>
  <pe:security-policies>
    <pe:security-policy name="optional_unique_name" override="composer_
security_policy_name">
      <pe:policy-class>fully_qualified_custom_policy_class_
name</pe:policy-class>
    </pe:security-policy>
  </pe:security-policies>
</pe:security-config>
```

   where,

   `override` specifies the default policy to be overridden. Options are `pagePolicy` and `taskflowPolicy`.

   For information about task flow policies, see Section 24.6, "Implementing Task Flow Security."

   Note: Out-of-the-box policies that are not mentioned here will continue to be checked by default.

   `<pe:policy-class>` specifies the name of the custom policy class.

3. Save the `adf-config.xml` file.

24.7.3 What Happens at Runtime

Composer performs the security checks defined in your custom policy and enables customization on components accordingly.

24.8 Troubleshooting Problems with Composer Components Security

This section provides information to assist you in troubleshooting security-related problems you may encounter while using Composer components.

For information about configuring logging, see "Configuring ADF Logging for Composer".

Problem

The `Show Detail Frame` and `Panel Customizable` do not show the **Edit** icon, or the `Panel Customizable` does not show the **Add Content** button. Also, you are not able to move the `Show Detail Frame` out of a `Panel Customizable` component or drop it into another `Panel Customizable` component.

Solution

Ensure that the `Panel Customizable` is not restricted by using MDS and the component actions are not secured by using entries in `adf-config.xml`.

Problem

You are not able to grant permissions on task flows in the `jazn-data.xml` file.

Solution

Ensure that you run the server with the Java startup parameter, as mentioned in Section 24.6, "Implementing Task Flow Security."

25

Customizing Task Flows

You can extend or alter the look and feel and functionality of Oracle WebCenter Portal task flows using the Oracle JDeveloper Customization Developer role. When you apply task flow customizations to a deployed Framework application, the customizations apply to all instances of that task flow in the application. You do not need to deploy customizations for individual task flow instances.

In Oracle WebCenter Portal: Spaces, task flow customizations are deployed at application-level, so any customizations that you make will apply to all spaces. To customize a taskflow for a specific space only, you must use runtime Spaces administration tools for taskflow customization as explained in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. If your customization involves minor modifications like adding text, hiding existing content or rearranging existing content, you can also use runtime administration tools. If your customization requires complex layout changes to be applied to all instances of a space, use the development-based customization approach explained in this chapter.

> **Note:** While you can perform view-level customizations to Framework applications, such as task flow customization, ADF model and Controller customizations are not supported in this release. To learn more about the different customization types, see the *Oracle Fusion Middleware Java EE Developer's Guide for Oracle Application Development Framework*.

This chapter includes the following sections

- Section 25.1, "Preparing for Task Flow Customization": Before you can customize task flows, you must enable your application to accept these customizations.
- Section 25.2, "Configuring the JDeveloper Customization Developer Role": After you have enabled customization for your application, you must configure the customization layer values to use in the JDeveloper Customization Developer role.
- Section 25.3, "Customizing Oracle WebCenter Portal Task Flows": The instructions in this section describe how to customize task flows in WebCenter Portal: Spaces or a WebCenter Portal: Framework application. The examples in this section demonstrate WebCenter Portal task flow customization using the JDeveloper Customization Developer role.
- Section 25.4, "Applying Task Flow Customizations to Deployed WebCenter Portal Applications": Once you complete a task flow customization, you must apply it to the deployed WebCenter Portal application by following the steps in this section.

- Section 25.5, "Removing Customizations from Deployed WebCenter Portal Applications": To revert task flow behavior or look and feel to the original deployment, remove the task flow customizations using the steps in this section.
- Section 25.6, "Catalog of WebCenter Portal Task Flows": This section lists WebCenter Portal task flows available for customization.

25.1 Preparing for Task Flow Customization

The steps below prepare WebCenter Portal applications for customizing task flows.

1. In JDeveloper, install the WebCenter Portal Customization JDeveloper Extension. To do so, choose **Help** > **Check for Updates**.
2. In the Check for Updates wizard, click **Next**.
3. On the Search for Updates page, select the **Oracle Fusion Middleware Products update center**, and click **Next**.
4. On the Updates page, in the Available Updates field, enter `webcenter`.
5. Select the **Oracle WebCenter Portal Customization Framework Design Time** extension and click **Finish**.

Note: If you have not already installed the latest version of the Oracle WebCenter Portal Framework and Services Design Time extension, you must install it before the customization extension.

25.1.1 WebCenter Portal: Spaces: Create a New Task Flow Customization Application

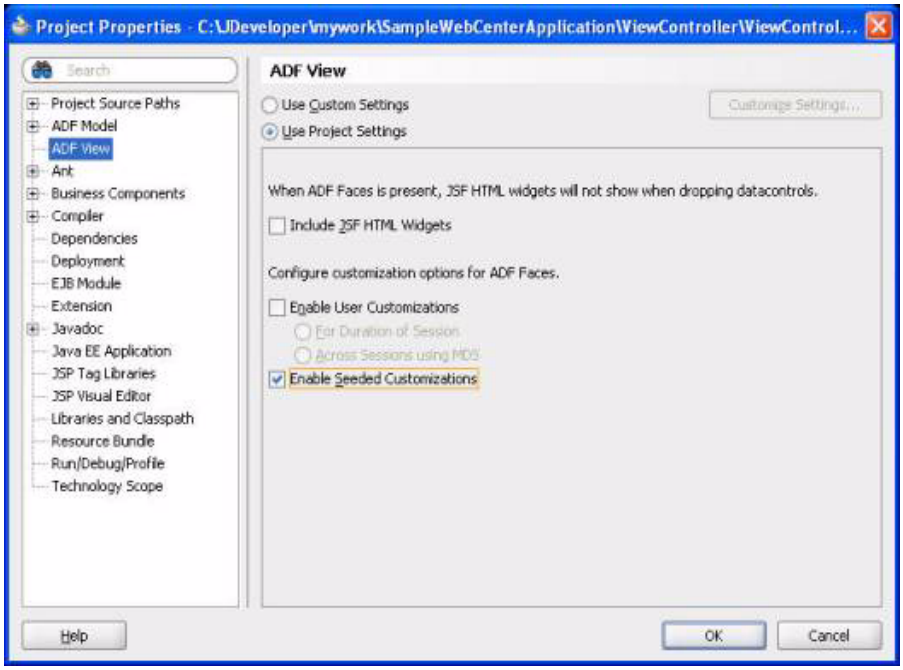
If you are customizing task flows in WebCenter Portal: Spaces, open JDeveloper and create a new WebCenter Spaces Task Flow Customization Application.

25.1.2 WebCenter Portal: Framework Application: Enable Customization

To enable task flow customization in your Framework application, first ensure that you have created your application using the WebCenter Portal: Framework Application template. (For information on creating a new Framework application, see Section 5.2, "Creating a Framework Application.") Then follow the steps below.

1. In the Application Navigator, right-click the **ViewController** project, and choose **Project Properties**.
2. In the Project Properties dialog, select **ADF View**, then select the **Enable Seeded Customizations** checkbox. Click **OK** and save your files.

Figure 25–1 Enabling Seeded Customizations

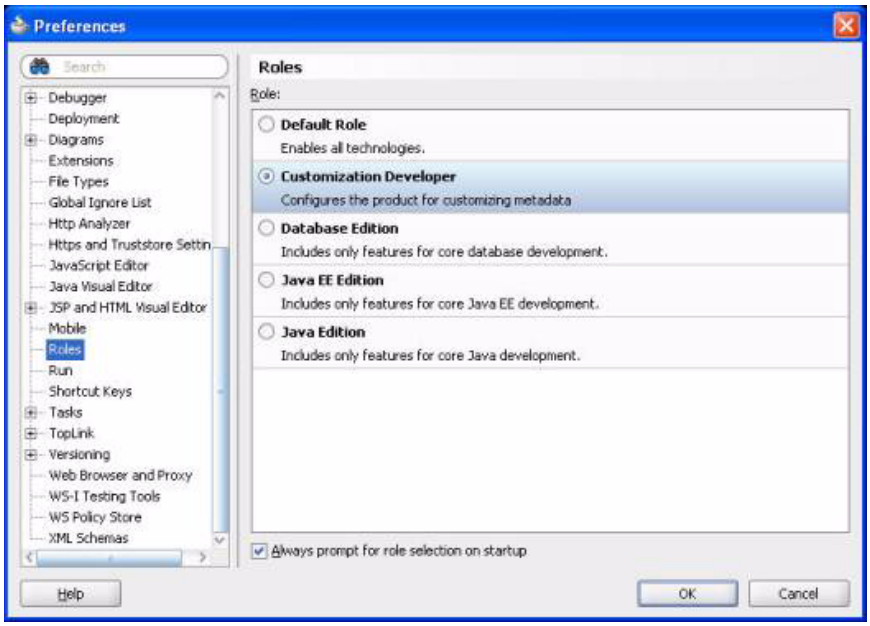


25.2 Configuring the JDeveloper Customization Developer Role

After you have enabled customization for your application as described in Section 25.1, "Preparing for Task Flow Customization", you must configure the customization layer values to use in the JDeveloper Customization Developer role.

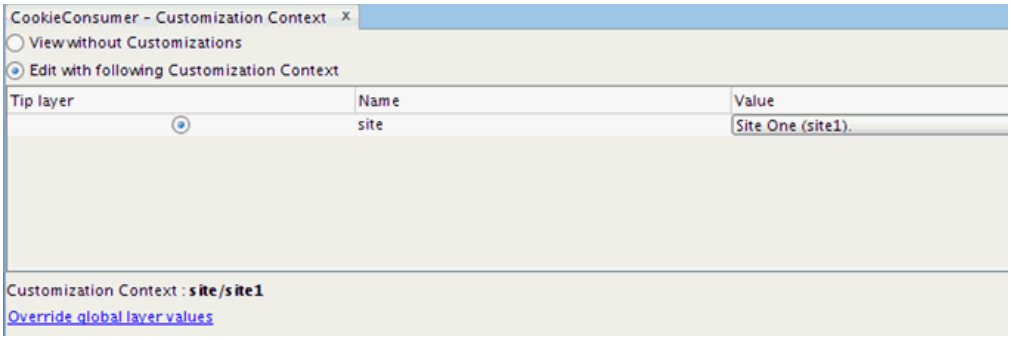
1. In JDeveloper, choose **Tools** > **Preferences** > **Roles**, select the **Customization Developer** role, then click **OK**.

Figure 25–2 Selecting the Customization Developer Role



2. JDeveloper prompts you to restart. Choose **Yes** to restart JDeveloper in the Customization Developer role.
3. For custom Framework applications, go to the **Customization Context** window and click **Override global layer values**.

Figure 25–3 Customization Context: Override Global Layer Values

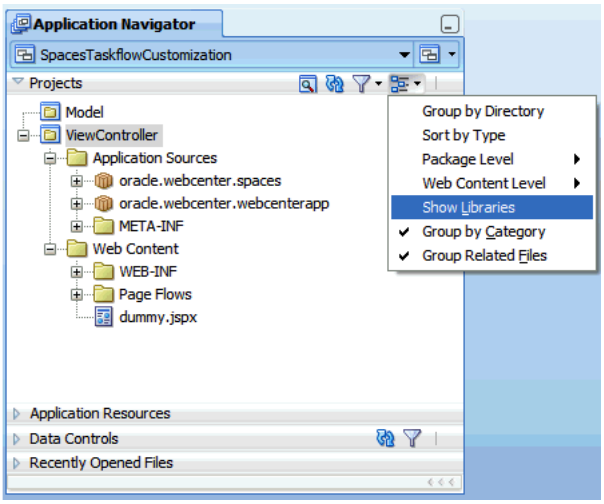


Enter the configuration for your application's customization class. For applications using the default ADF site customization class, use the following code:

```
<cust-layers xmlns="http://xmlns.oracle.com/mds/dt">
  <cust-layer name="site" id-prefix="s">
    <cust-layer-value value="site" display-name="Site" id-prefix="s"/>
  </cust-layer>
</cust-layers>
```

4. Ensure that **Show Libraries** is enabled in the dropdown (on the far right) of the **Projects** bar in JDeveloper navigator.

Figure 25–4 Navigator Display Options - Show Libraries



In the Customization Developer role, JDeveloper will display a red man next to your application name in the Application Navigator. You will also see a Customization Context window with **Edit with following Customization Context** selected and the customization layer name selected as `site`, and the layer value set to the value defined in `CustomizationLayerValues.xml`. These values indicate that you have successfully configured your application to enable customization of WebCenter Portal task flows.

25.3 Customizing Oracle WebCenter Portal Task Flows

JDeveloper's Customization Developer role is a powerful mechanism that allows you to customize the ADF Library without changing the code in base library JAR. Since both Oracle WebCenter Portal and Oracle ADF leverage MDS, you can use the Customization Developer role to extend WebCenter Portal: Spaces and WebCenter Portal: Services task flows. All WebCenter Portal task flows are packaged in the ADF Library so task flow customization is possible in JDeveloper design time. For instructions on configuring the Customization Developer role, see Section 25.2, "Configuring the JDeveloper Customization Developer Role"

The use cases that can be achieved by task flow customization include:

- Change labels including column names for task flows.
- Change icons used in task flows, for example replacing the Edit and Refresh icons with new icons on the Discussions - Space Forums task flow.
- Change the order of table columns in a task flow.
- Remove an entire region or component, such as the Search box from primary tab.
- Show additional attributes using Expression Language, for example customizing the People Picker task flow to show an additional user profile attribute.

After you have prepared your application for customization as described in the previous sections, open the application and make any necessary customizations.

- For examples of task flow customization, see Section 25.3.1, "Examples: Customizing Task Flows for WebCenter Portal: Spaces" and Section 25.3.2, "Examples: Customizing Task Flows for Custom Framework Applications."
- For a list of customizable task flows, see Section 25.6, "Catalog of WebCenter Portal Task Flows."
- After you have completed your customizations, deploy them as explained in Section 25.4, "Applying Task Flow Customizations to Deployed WebCenter Portal Applications."

25.3.1 Examples: Customizing Task Flows for WebCenter Portal: Spaces

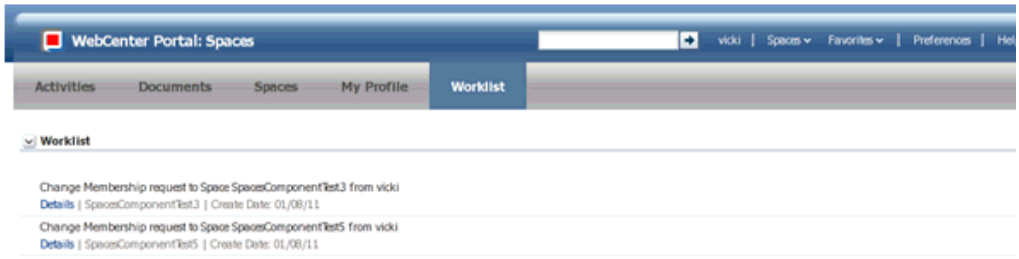
To customize task flows in WebCenter Portal: Spaces, open the application you created in Section 25.1.1, "WebCenter Portal: Spaces: Create a New Task Flow Customization Application" and make your customizations. The examples that follow explain two common customizations:

- Section 25.3.1.1, "Example: Customizing the Worklist Task Flow"
- Section 25.3.1.2, "Example: Customizing the Discussion Forums Task Flow"

25.3.1.1 Example: Customizing the Worklist Task Flow

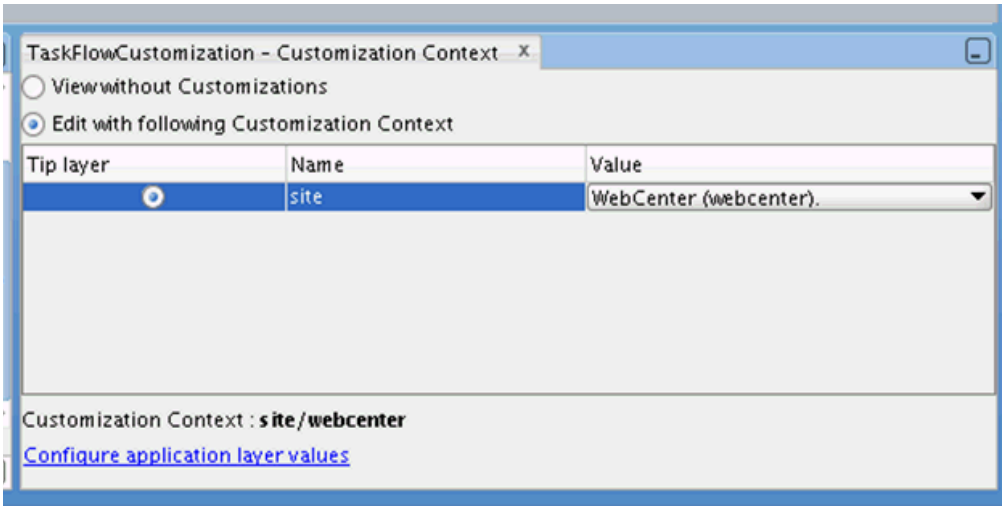
This example shows how to customize the Spaces Worklist task flow by replacing the two-row entry for each worklist item with a single row that provides a link to the worklist item details with a popup that displays the details that were previously shown on the second row. The following image shows a sample Worklist with the default two-row entry configuration.

Figure 25–5 Worklist - Before Customization



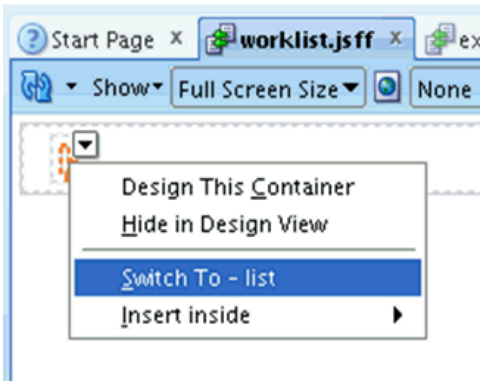
1. If you are not already using the Customization Developer role, switch to this role (choose **Tools** > **Preferences**, select **Role**, then select **Customization Developer Role** and restart JDeveloper).
2. Open your WebCenter Portal: Spaces Task Flow Customization Application.
3. In the Application Navigator, expand the **ViewController** project to view the **WebCenter Worklist Service View** library. (If you cannot see the library, right-click the ViewController project, choose Navigation Display Options in the Projects bar, and select Show Libraries.)
4. Ensure that the **Customization Context** panel is configured so that edits will be done with the WebCenter customization layer applied, as shown below.

Figure 25–6 Customization Context



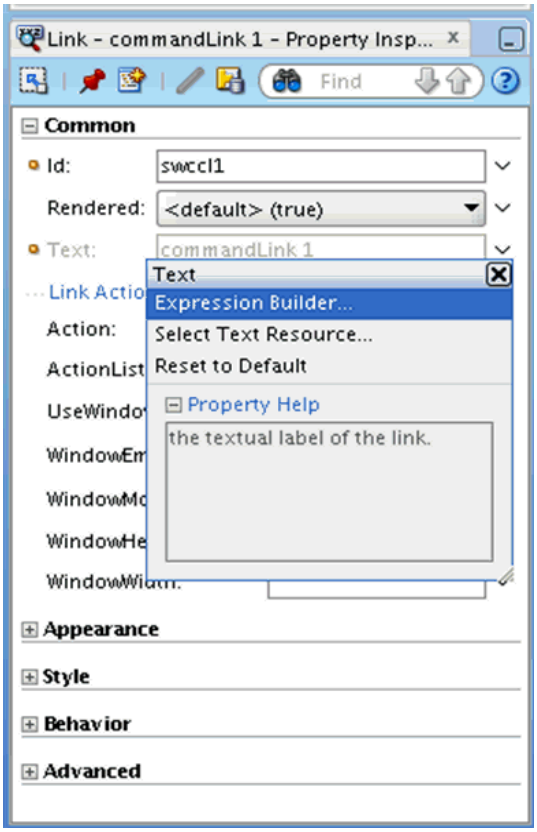
5. In the WebCenter Worklist Service View library open the **worklist.jsff** file under oracle.webcenter.worklist.view.jsf.fragments.
6. Switch to the list view facet of the view. Click the empty facet on the page and choose **Switch To - list**.

Figure 25–7 Switch To List View



7. In the Structure window, expand the structure tree to locate the `af:outputText - #{row.title}` component. Right-click the component and and choose **Insert Before af:outputText - #{row.title}** > **Link**.
8. In the Structure window, click on the newly created `af:commandLink` to open the Property Inspector dialog.
9. In the Property Inspector, click the down arrow next to the **Text** field and choose **Expression Builder**. Enter `#{row.title}` as the expression.

Figure 25–8 Property Inspector



10. Expand the Appearance section of the Property Inspector. Click the down arrow next to the **ShortDesc** field and choose **Expression Builder**. Enter `#{row.dateInfoSummary}` as the expression.
11. In the **ActionListener** field, enter `openTaskDetailsApp()`
12. Find the newly added `commandLink` and delete the following components:
    `af:outputText - #{row.title}`
    `af:panelGroupLayout`
13. The worklist.jsff.xml file that is generated (under the ADF Library Customizations node in the Application Navigator) should include code similar to the following:

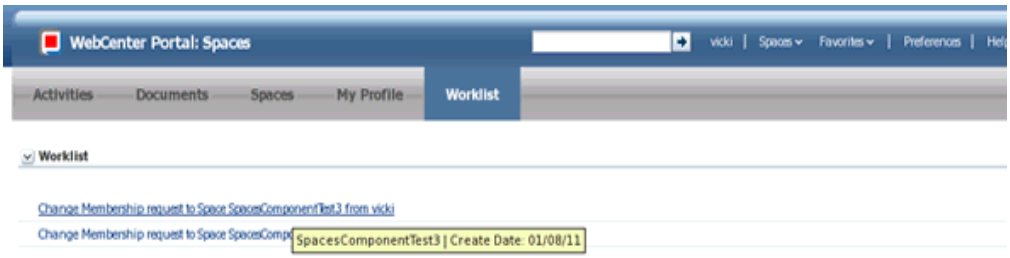
Example 25–1 Updated worklist.jsff file

```
<mds:customization version="11.1.1.61.15"
                   xmlns:mds="http://xmlns.oracle.com/mds">
  <mds:insert parent="pg13" position="first
              xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
    <af:commandLink xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
                text="#{row.title}" id="swccl1"
                shortDesc="#{row.dateInfoSummary}"
                actionListener="#{backingBeanScope.wlbbean.openTaskDetailsApp}"/>
  </mds:insert>
  <mds:replace node="ot2"/>
  <mds:replace node="pg14"/>
</mds:customization>
```

14. Deploy the metadata deployment profile to the WebCenter Portal: Spaces application as explained in Section 25.4, "Applying Task Flow Customizations to Deployed WebCenter Portal Applications."

The image below shows a sample Worklist after customization, with a single row and a popup.

Figure 25–9 Worklist - After Customization



25.3.1.2 Example: Customizing the Discussion Forums Task Flow

This example shows how you can customize the WebCenter Portal: Spaces Discussion Forums task flow to display a profile image for the user who initiated the discussion. The following image shows a sample Discussion Forum without the profile image.

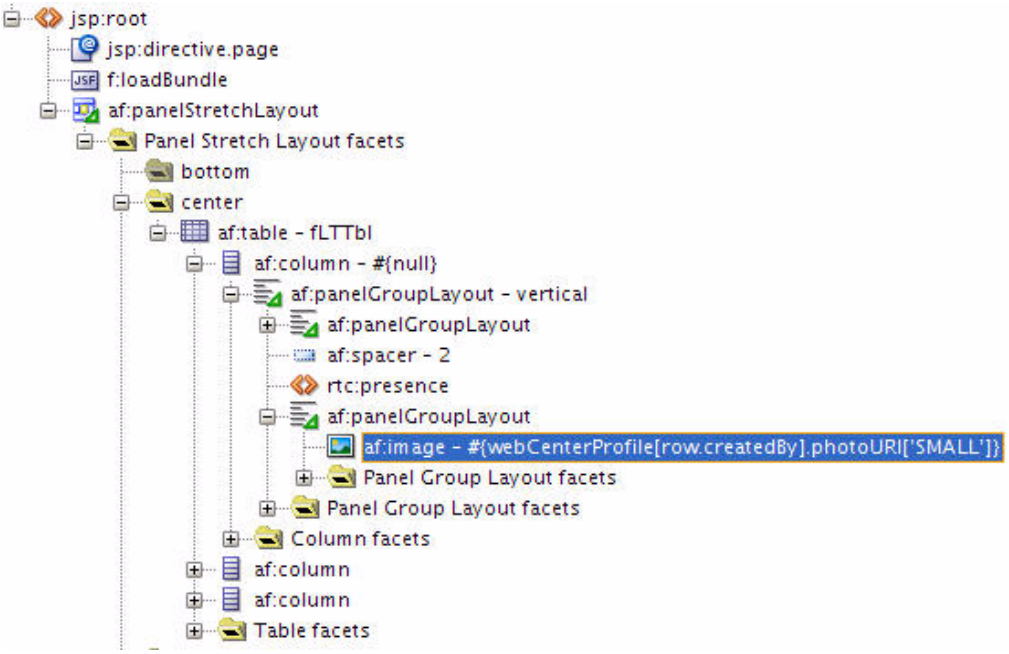
Figure 25–10 Discussion Forum - Before Customization



To customize the Discussion Forums task flow:

1. If you are not already using the Customization Developer role, switch to this role (choose **Tools** > **Preferences**, select **Role**, then select **Customization Developer Role** and restart JDeveloper).
2. Open your WebCenter Portal: Spaces Task Flow Customization Application.
3. In the Application Navigator, expand the **ViewController** project to view the **WebCenter Portal Discussion Services View** library. (If you cannot see the library, right-click the ViewController project, choose Navigation Display Options in the Projects bar, and select Show Libraries.)
4. In the WebCenter Portal Discussion Services View library, open the **ListTopics.jsff** file under oracle.webcenter.collab.forum.view.jsf.fragments.
5. In the Structure window, search for the `<rtc:presence>` tag on the page. This tag renders the user name.

Figure 25–11 Structure Window



6. Right-click the `rtc.:presence` tag, then choose **Insert After** > **ADF Faces** and select **Panel Group Layout** from the dialog.

7. In the Structure window, right-click the new **Panel Group Layout** and choose **Insert** > **Image Component**.
8. In the Property Inspector for the new Image component, set the Source property to `#{webCenterProfile[row.createdBy].photoURI['SMALL']}`. This EL will return the location of the image that the current user has set as their profile photo.
9. Save your files and rebuild the project.

   The ListTopics.jsff.xml file that is generated will contain the following code:

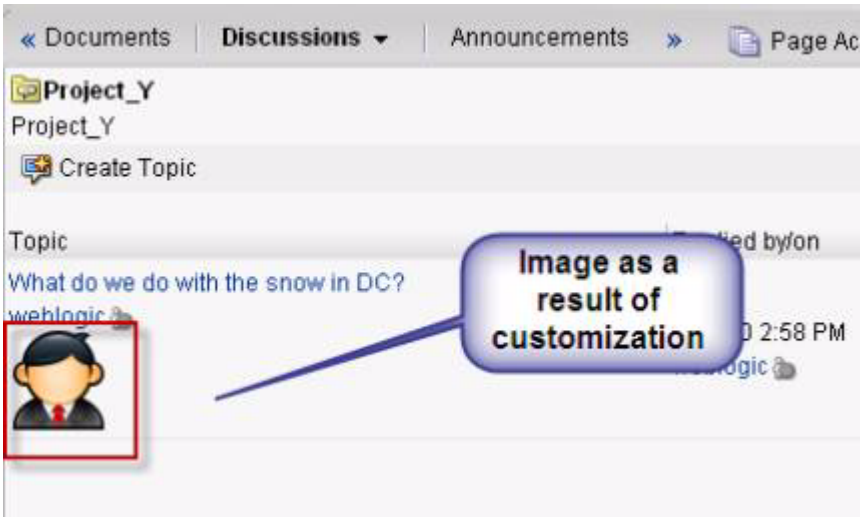
Example 25–2 Updated ListTopics.jsff.xml File

```
<mds:customization version="11.1.1.60.46" xmlns:mds="http://xmlns.oracle.com/mds">
 <mds:insert after="p1" parent="frmltpgl11"
             xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
    <af:panelGroupLayout xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
                    id="swcpgl1">
      <af:image xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
        source="#{webCenterProfile[row.createdBy].photoURI['SMALL']}" id="swci1"/>
    </af:panelGroupLayout>
  </mds:insert>
</mds:customization>
```

10. Deploy the metadata deployment profile to the Spaces application as explained in Section 25.4, "Applying Task Flow Customizations to Deployed WebCenter Portal Applications."

The following image shows a sample Discussion Forum after customization, with a profile image.

Figure 25–12 Discussion Forum - After Customization (With User Profile Image)



25.3.2 Examples: Customizing Task Flows for Custom Framework Applications

To customize task flows in a custom Framework application, first follow the instructions in Section 25.1.2, "WebCenter Portal: Framework Application: Enable Customization." The example that follows demonstrates a common customization.

25.3.2.1 Example: Customizing the Document Library - List View Task Flow

The example that follows shows how to customize the Document Library – List View task flow to add a new column called HTML Rendition. Before you perform these steps, ensure that you have prepared a customizable Framework application application as described in Section 25.1.2, "WebCenter Portal: Framework Application: Enable Customization.".

1. Open your custom Framework application in JDeveloper and confirm that you are using the Customization Developer role.
2. In the Resource Palette, under My Catalogs, expand the WebCenter Portal Services Catalog and open the Task Flows folder.
3. Right-click **Document Library – List View** and choose **Add to project**. If a confirmation dialog displays, click **Add Library**.
4. In the Application Navigator, expand the **ViewController** project to view the **WebCenter Document Library Services View** library. (If you cannot see the library, right-click the ViewController project, choose Navigation Display Options in the Projects bar, and select Show Libraries.)
5. In the WebCenter Portal Document Library Services View library, open the **docListViewerTableTemplate.jsff** file under oracle.webcenter.doclib.view.jsf.fragments.
6. Add a new column after `ITEM_NAME_COLUMN_HEADER`.
   a. In the Structure window, right-click the column with title `#{dlBndl.ITEM_NAME_COLUMN-HEADER}` and choose Copy and Paste to create a new ADF column component.
   b. In the Property Inspector for the new ADF column component, change the `headerText` property to "HTML Rendition".
   c. In the Structure window, expand the new column and delete the `af:image` component.
   d. Right click the `af:goLink` component inside `af:switcher -> f:facet – false` and choose **Go to property**.
   e. Change the values of the Text and Destination. For example:

```
Text=" Get Conversion (HTML)"
Destination="http://host:port/idcplg?IdcService=GET_DYNAMIC_CONVERSION
&amp;dID=#{item.dID}&amp;coreContentOnly=1#{item.portletLinkPostfix}"
```

7. Save your files.

25.4 Applying Task Flow Customizations to Deployed WebCenter Portal Applications

The previous sections explained how to customize WebCenter Portal task flows for different scenarios. The output of these exercises is the generated MDS customization. The customizations show up as `.xml.xml` or `.jsff.xml` files in the View project of the application under the `libraryCustomization` package. These customization documents are essentially instructions for MDS to apply changes on top of the base document that is shipped to show the customized behavior at runtime.

Once you complete a task flow customization, you must apply it to the deployed WebCenter Portal application (Framework or Spaces). To see customizations

performed on task flows in JDeveloper at runtime, you must import these customizations to the MDS repository of the deployed application.

> **Note:** This process updates the runtime WebCenter Portal application metadata repository; back up the MDS schema before performing these steps. Also, it is a best practice to test your customizations in a development or stage environment first.

To import WebCenter Portal task flow customizations to the MDS repository, use the steps in the following sections:

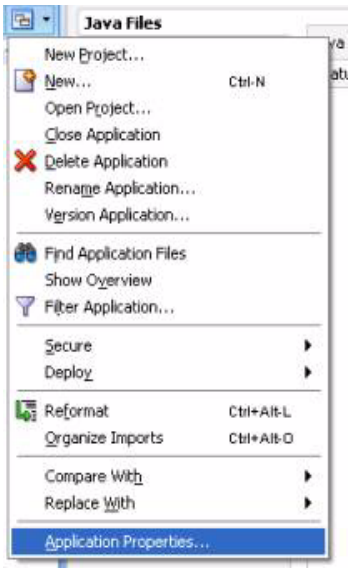
- Section 25.4.1, "Creating a Metadata Deployment Profile for Custom WebCenter Portal Applications"
- Section 25.4.2, "Deploying Task Flow Customizations Directly from JDeveloper"
- Section 25.4.3, "Deploying Task Flow Customizations Using WLST"

25.4.1 Creating a Metadata Deployment Profile for Custom WebCenter Portal Applications

Use the steps that follow to create a metadata deployment profile for your custom WebCenter Portal application. (These steps are not necessary if you are using the WebCenter Portal: Spaces Task Flow Customization Application.)

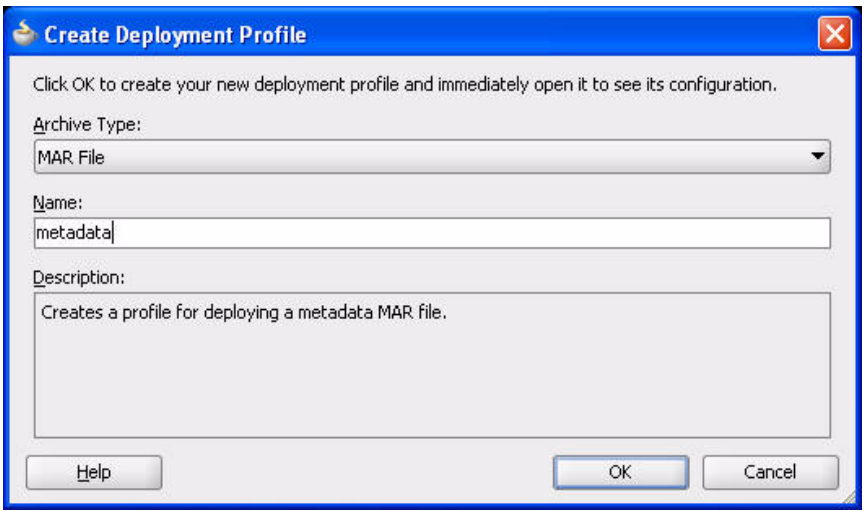
1. In the Application Navigator, from the Application Menu, choose **Application Properties**.

Figure 25–13 Application Properties



2. Choose **Deployment** and create a deployment profile by using the MAR File archive type.

Figure 25–14 Creating a Deployment Profile



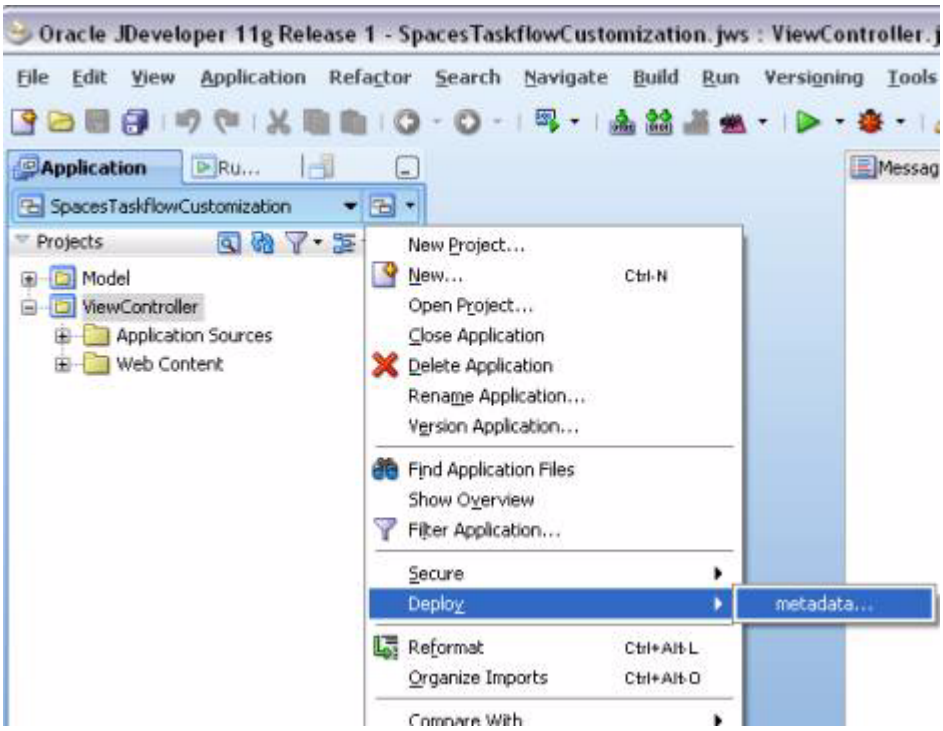
3. Click **OK** to close the Create Deployment Profile dialog.
4. Click **OK**, then click **OK** again to finish creating the Deployment Profile.

25.4.2 Deploying Task Flow Customizations Directly from JDeveloper

Use the steps that follow to deploy your task flow customizations directly from JDeveloper.

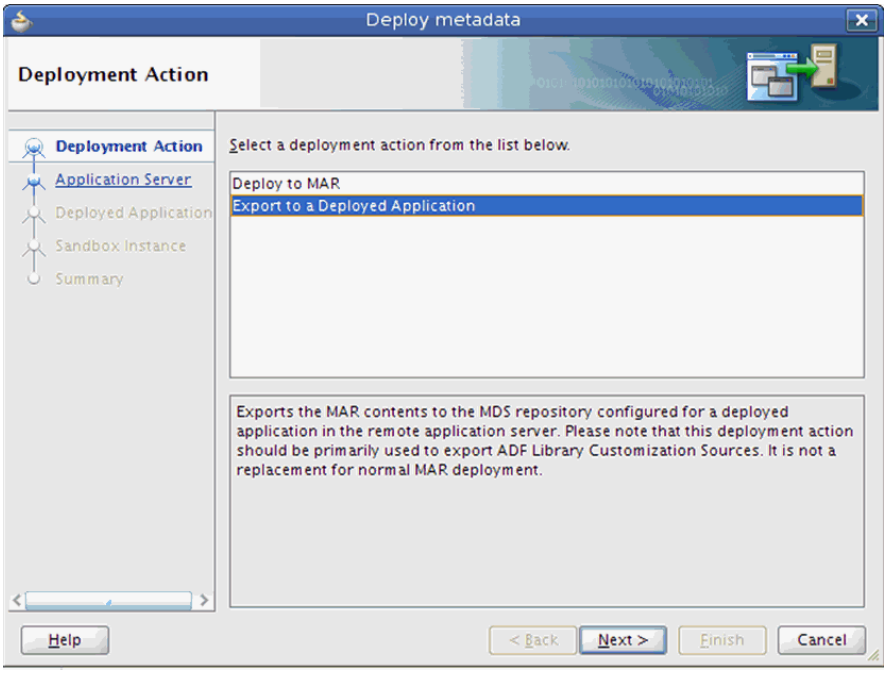
1. In the Application Navigator, from the Application Menu, choose **Deploy** > **metadata** (`metadata` is the name of the Deployment Profile).

Figure 25–15 Deploying the "metadata" Deployment Profile



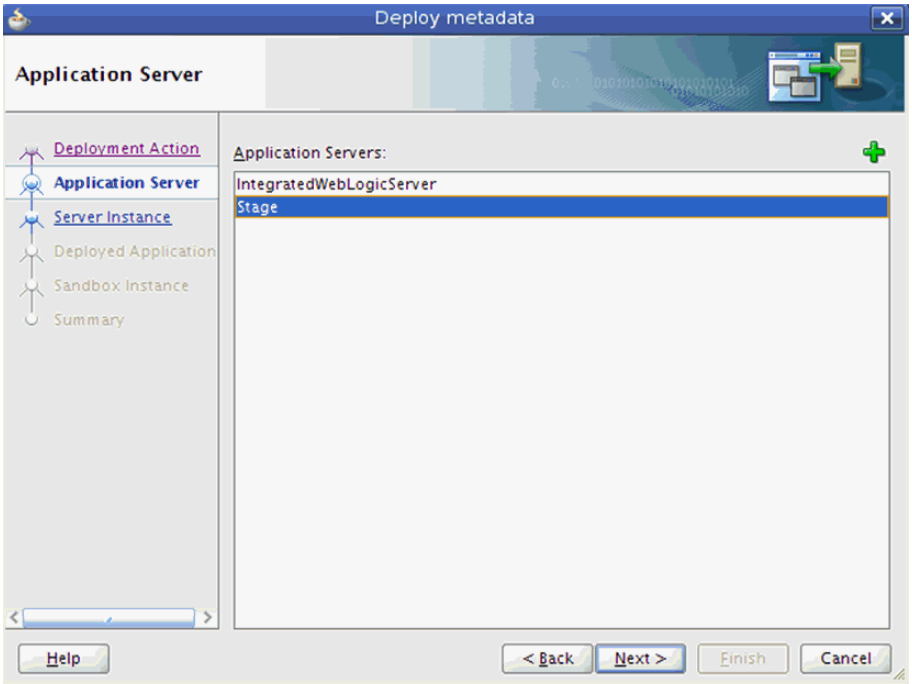
2. In the **Deploy metadata** dialog, select **Export to a Deployed Application** and click **Next**.

Figure 25–16 Deploy Metadata Dialog



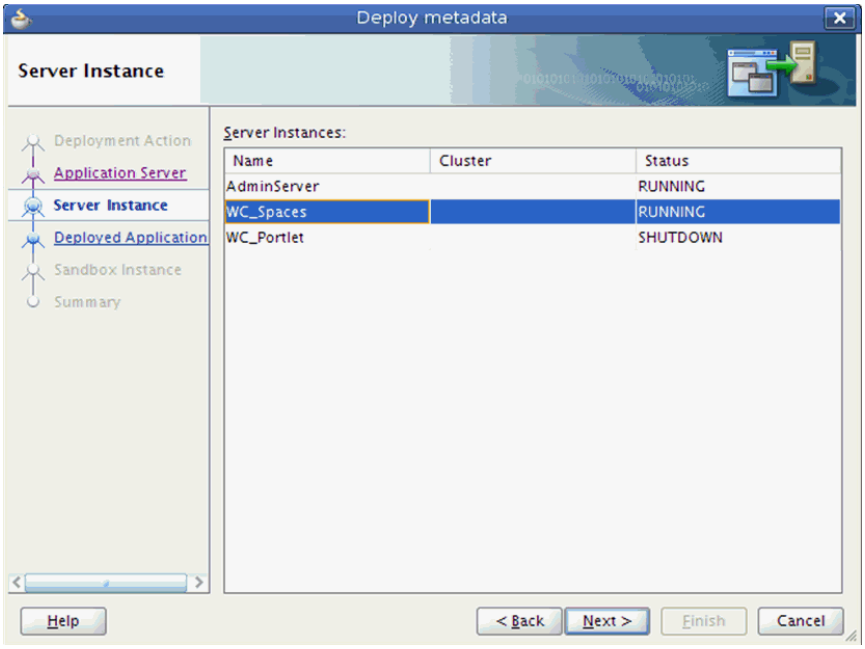
3. On the **Application Server** page, select the application server connection for the instance to which you are deploying or create a new connection by clicking the "+" icon. Click **Next**.

Figure 25–17 Deploy Metadata: Application Server



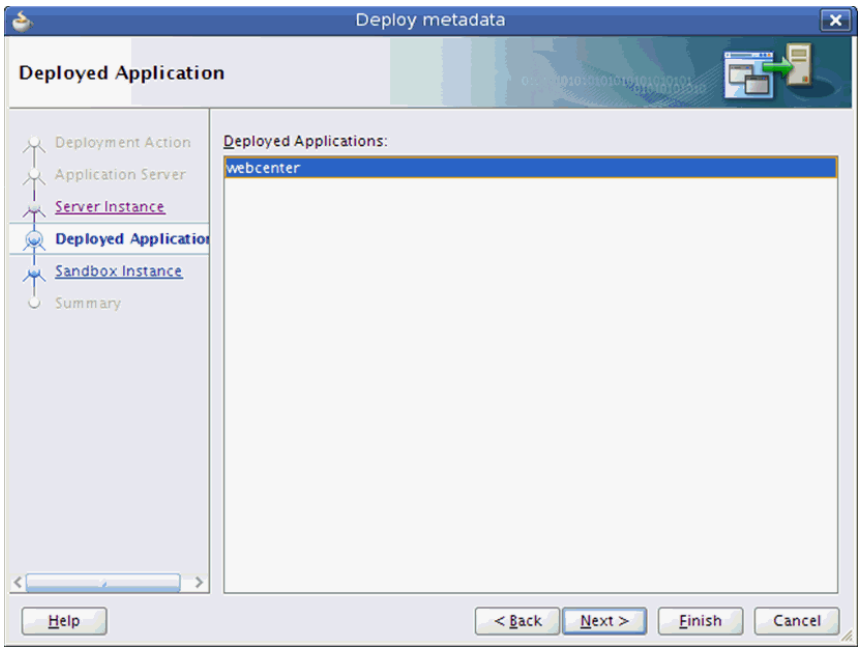
4. On the **Server Instance** page, select the server on which the application you are customizing is deployed. For a high availability installation with multiple servers hosting a single application, the customizations only need to be deployed to one server. Click **Next**.

Figure 25–18 Deploy Metadata Dialog: Server Instance



5. On the **Deployed Application** page, select the application to which the customizations should be deployed. Click **Next**.

Figure 25–19 Deploy Metadata: Deployed Applications



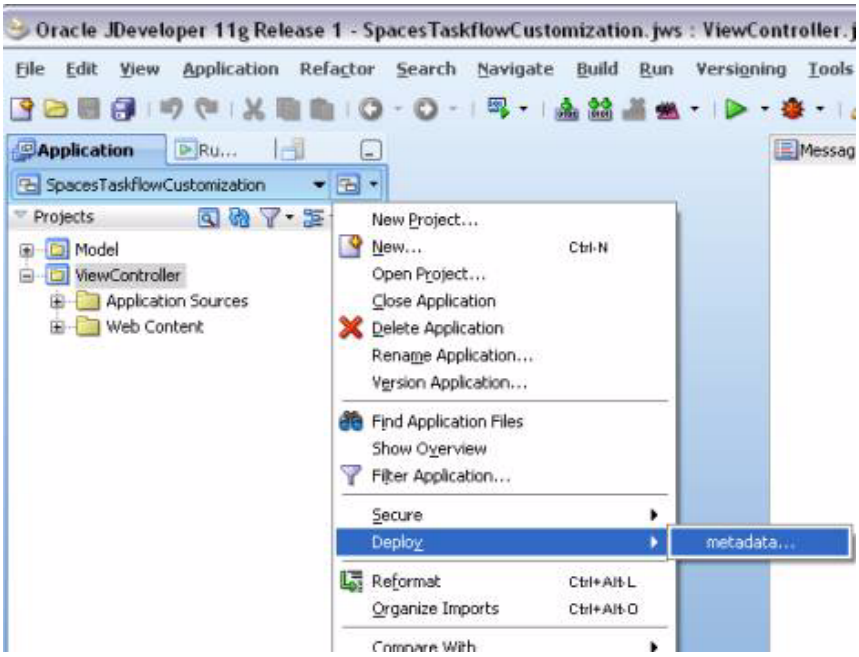
6. Click **Finish** to deploy the customization.

25.4.3 Deploying Task Flow Customizations Using WLST

Use the steps that follow to deploy task flow customizations using WLST.

1. In the Application Navigator, from the Application Menu, choose **Deploy** > **metadata** (`metadata` is the name of the Deployment Profile).

Figure 25–20 Deploying the "metadata" Deployment Profile



2. In the Deploy metadata dialog, choose Deploy to MAR file, then click **Finish**.

   Tip: Alternatively, you can export the metadata to a deployed application by selecting the **Export to Deployed Application** option, configuring the connection details of the server on which the application is deployed, and selecting the appropriate application.

3. JDeveloper creates the MAR file in the application directory of your *JDEV_USER* home directory. Copy this MAR file, which contains the customizations, to the directory where the application is deployed. Maintain a backup of this file so that you always have a version of the customizations for this application.
4. Use the MDS WLST import command to import the task flow customizations to the application's MDS repository.

Example 25–3 Command Line for Importing Customizations

```
wls:/weblogic/serverConfig>importMetadata(application='webcenter',
server='WC_Spaces',
fromLocation='/path/metadata.mar',
docs='/**')
```

For more information on MDS WLST commands, refer to the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference* on the Oracle Technology Network.

25.5 Removing Customizations from Deployed WebCenter Portal Applications

You can revert task flow behavior or look and feel to the original deployment by removing task flow customizations.

Use the MDS WLST `deleteMetadata` command to remove the applied customizations. The `deleteMetadata` command needs to exercise with caution as incorrect use of this command may cause the loss of metadata documents. The sample command below removes the customization created in Section 25.3.1.2, "Example: Customizing the Discussion Forums Task Flow."

Example 25–4 Command to Delete the Customization Metadata from a Deployed WebCenter Portal Application

```
deleteMetadata(application='webcenter',server='WC_Spaces',
docs='/oracle/webcenter/webcenterapp/view/taskflows/discussionServices/**',
excludeBaseDocs='true')
```

For more details on MDS WLST commands, refer to the *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference* on the Oracle Technology Network.

25.6 Catalog of WebCenter Portal Task Flows

The following task flows have been validated for view-level customizations. You can find these task flows in the libraries or JAR files and definition paths specified in Table 25–1.

Note: To customize analytics task flows, you must add the following JARs to the project:

JDEV_HOME`/webcenter/modules/oracle.webcenter.framework_11.1.1/analytics-reporting-service-view.jar`

JDEV_HOME`/webcenter/modules/oracle.webcenter.framework_11.1.1/analytics-reporting-service-model.jar`

Table 25–1 WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
adfpage-renderer	WebCenter Portal: Framework	`oracle/webcenter/portal/framework/navigation/renderer/adfpage-renderer.xml`
Activity Stream - Mini View	WebCenter Portal Activity Streaming Service View	`oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-miniview.xml`
Activity Stream - Main View	WebCenter Portal Activity Streaming Service View	`oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-mainview.xml`
Analytics - Console	Task flows included in `analytics-reporting-service-view.jar` and `analytics-reporting-service-model.jar`	`oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-console.xml`

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Analytics - WebCenter Portal Traffic	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-summary.xml
Analytics - Page Traffic	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-page-traffic.xml
Analytics - Login Metrics	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-login.xml
Analytics - Portlet Traffic	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-portlet-traffic.xml
Analytics - Portlet Response Time	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-portlet-resptime.xml
Analytics - Portlet Instance Traffic	Task flows included analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-portlet-instance-traffic.xml
Analytics - Portlet Instance Response Time	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-portlet-instance-resptime.xml
Analytics - Search Metrics	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-search-phrases.xml
Analytics - Document Metrics	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-document-view.xml
Analytics - Wiki Metrics	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-wiki-view.xml
Analytics - Blog Metrics	Task flows included in analytics-reporting-service-view.jar and analytics-reporting-service-model.jar	oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-blog-view.xml

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Analytics - Discussion Metrics	Task flows included in `analytics-reporting-service-view.jar` and `analytics-reporting-service-model.jar`	`oracle/webcenter/analytics/view/jsf/taskflows/seeded/analytics-report-discussion-forum-view.xml`
Announcements	WebCenter Portal Announcement Service View	`oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml`
Announcements - Quick View	WebCenter Portal Announcement Service View	`oracle/webcenter/collab/announcement/view/taskflows/mini-view-definition.xml`
Application Navigator	WebCenter Portal: Spaces View	`oracle/webcenter/webcenterapp/view/taskflows/applinks/AppNavigatorRegion.xml`
Blogs		`oracle/webcenter/blog/view/jsf/taskflows/blogDigestViewer/blog-main-view.xml`
Calendar Main View	WebCenter Portal Events Service View	`oracle/webcenter/collab/events/view/taskflows/calendar-main-view.xml`
Calendar Mini View	WebCenter Portal Events Service View	`oracle/webcenter/collab/events/view/taskflows/calendar-mini-view.xml`
ChooseLanguageTaskflow	WebCenter Portal: Spaces View	`oracle/webcenter/webcenterapp/view/taskflows/translations/ChooseLanguageTaskflow.xml`
CommunityBrowserRegion	WebCenter Portal: Spaces View	`oracle/webcenter/community/view/taskflows/browser/CommunityBrowserRegion.xml`
community-contacts-task-flow	WebCenter Portal: Spaces View	`oracle/webcenter/people/view/jsf/regions/community-contacts-task-flow.xml`
Connections - Card	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/table-of-connections-taskflow.xml`
Connections - Detailed View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-advanced-taskflow.xml`
Connections - Main View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-taskflow.xml`
Connections - Main View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-untabbed-taskflow.xml`
Connections - Mini View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-taskflow.xml`
Connections - Network	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-network-taskflow.xml`
content-renderer	WebCenter Portal: Framework	`oracle/webcenter/portal/framework/navigation/renderer/content-renderer.xml`
customization-manager-taskflow	Oracle Composer	`oracle/adfinternal/pageeditor/manager/taskflows/customization-manager-taskflow.xml`
Discussions	WebCenter Portal Discussions Service View	`oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml`

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Discussions - Manager	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/main-task-flow.xml
Discussions - Popular Topics	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/popularTopic-task-flow.xml
Discussions - Recent Topics	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/recentTopic-task-flow.xml
Discussions - Space Forums	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml
Discussions - Quick View	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml
Discussions - Watched Forums	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/watchedForum-task-flow.xml
Discussions - Watched Topics	WebCenter Portal Discussions Service View	oracle/webcenter/collab/forum/view/taskflows/watchedTopic-task-flow.xml
Documents - AutoVue	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/docViewer/autovue.xml
Documents - Content Presenter	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/presenter/contentPresenter.xml
Documents - Document Manager	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml
Documents - Document Navigator	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/treeNav/treeNavigator.xml
Documents - Document Viewer	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/docViewer/documentViewer.xml
Documents - Folder Viewer	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/folderViewer/folderView.xml
Documents - Document Explorer	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/explore/explorer.xml
Documents - List Viewer	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/docListViewer.xml
Documents - Main View	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml
Documents - Mini Properties	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/miniProperties/miniProps.xml
Documents - Properties	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/docViewer/docInfo.xml
Documents - Recent Documents	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/recentDocuments.xml
Documents - Rich Text Editor	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/richTextEditor/editor.xml
Documents - Upload	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/upload/uploader.xml
Documents - Version History	WebCenter Portal Document Library Service View	oracle/webcenter/doclib/view/jsf/taskflows/versionHistory/history.xml
Events	WebCenter Portal Events Service View	oracle/webcenter/collab/events/view/taskflows/calendar-main-view.xml

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Exportregion	Internal - invoked by WebCenter Portal: Spaces administrator for export of spaces or space templates.	`oracle/webcenter/webcenterapp/view/taskflows/lifecycle/Exportregion.xml`
Feedback	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosDetailViewer.xml`
Feedback - Summary View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosMiniViewer.xml`
ImportRegion	Internal - invoked by WebCenter Portal: Spaces administrator for import of spaces or space templates.	`oracle/webcenter/webcenterapp/view/taskflows/lifecycle/ImportRegion.xml`
LifecycleCartTaskflow	Internal	`oracle/webcenter/webcenterapp/view/taskflows/lifecycle/LifecycleCartTaskflow.xml`
LifecycleSelectorTaskflow	Internal	`oracle/webcenter/webcenterapp/view/taskflows/lifecycle/LifecycleSelectorTaskflow.xml`
lifecycle-siteresource-export-task-flow	Internal - used in Resource Manager	`oracle/webcenter/lifecycle/view/jsf/regions/lifecycle-siteresource-export-task-flow.xml`
lifecycle-siteresource-import-task-flow	Internal - used in Resource Manager	`oracle/webcenter/lifecycle/view/jsf/regions/lifecycle-siteresource-import-task-flow.xml`
Links	WebCenter Portal Links Service View	`oracle/webcenter/relationship/view/jsf/resources/links-detail.xml`
Links Dialog	WebCenter Portal Links Service View	`oracle/webcenter/relationship/view/jsf/resources/links-detail-popup.xml`
List - Main View	WebCenter Portal List Service View	`oracle/webcenter/list/view/jsf/regions/main-view-task-flow.xml`
list-instance-view-task-flow	WebCenter Portal Lists Service View	`oracle/webcenter/list/view/jsf/regions/list-instance-view-task-flow.xml`
list-of-resource-types-taskflow		`oracle/webcenter/portalframework/srm/controller/taskflows/list-of-resource-types-taskflow.xml`
Mail	WebCenter Portal Mail Service View	`oracle/webcenter/collab/mail/view/jsf/regions/mini-view-definition.xml`
Message Board - Mini View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.xml`
Message Board - Main View	WebCenter Portal PeopleConnections View	`oracle/webcenter/peopleconnections/wall/controller/taskflows/WallDetailViewer.xml`
Navigation - Menu	Navigation Task Flows	`oracle/webcenter/navigationtaskflows/view/pagemenu-definition.xml`
Navigation - Breadcrumb	Navigation Task Flows	`oracle/webcenter/navigationtaskflows/view/pagebreadcrumb-definition.xml`
Navigation - Tree	Navigation Task Flows	`oracle/webcenter/navigationtaskflows/view/pagetree-definition.xml`

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
navigation-renderer	WebCenter Portal: Framework	oracle/webcenter/portal/framework/navigation/renderer/navigation-renderer.xml
Organization View	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/profile/view/jsf/regions/orgview/organization-view.xml
Page - Create New	WebCenter Portal Page Service View	oracle/webcenter/page/view/jsf/fragments/page-create-page.xml
Polls - Polls Manager	WebCenter Portal Polls and Surveys Service View	oracle/webcenter/collab/survey/view/jsf/taskflows/list-surveys-definition.xml
Polls - Quick View	WebCenter Portal Polls and Surveys Service View	oracle/webcenter/collab/survey/view/jsf/taskflows/quick-poll-definition.xml
Polls - Take Poll	WebCenter Portal Polls and Surveys Service View	oracle/webcenter/collab/survey/view/jsf/taskflows/take-polls-definition.xml
Polls - View Poll Results	WebCenter Portal Polls and Surveys Service View	oracle/webcenter/collab/survey/view/jsf/taskflows/view-results-definition.xml
portlet-renderer	WebCenter Portal Portal Framework	oracle/webcenter/portal/framework/navigation/renderer/portlet-renderer.xml
Presence Controls	WebCenter Portal IM and Presence Service View	oracle/webcenter/collab/rtc/view/jsf/regions/presence-controls-taskflow.xml
Profile - Actions	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/view/jsf/regions/profile-actions.xml
Profile Gallery	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/personalweb/view/jsf/regions/profile-gallery.xml
Profile - Main View	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/profile/view/jsf/regions/extended/extended-profile.xml
Profile - Snapshot	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml
Profile - Summary	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/profile/view/jsf/regions/summary/profile-summary.xml
Publisher	WebCenter Portal PeopleConnections View	oracle/webcenter/peopleconnections/wall/controller/taskflows/Publisher.xml
Recommended Connections	WebCenter Portal Activity Graph Service View	view/oracle/webcenter/activitygraph/controller/taskflows/recommended-connections.xml
Resource Action Handler - Resource Viewer	WebCenter Portal Common View	oracle/webcenter/framework/service/controller/taskflows/resourceViewer.xml
Recent Activities	WebCenter Portal Recent Activity Service View	oracle/webcenter/recentactivity/controller/taskflows/recent-activities.xml
RSS Viewer	WebCenter Portal RSS Service View	oracle/webcenter/rssviewer/view/jsf/fragments/RSSViewerTaskFlow.xml
Search	WebCenter Portal Search Services View	oracle/webcenter/search/controller/taskflows/searchResults.xml
Search Preferences	WebCenter Portal Search Services View	oracle/webcenter/search/controller/taskflows/preferences.xml

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Search - Saved Searches	WebCenter Portal Search Services View	`oracle/webcenter/search/controller/taskflows/allSavedSearches.xml`
Search Toolbar	WebCenter Portal Search Services View	`oracle/webcenter/search/controller/taskflows/localToolbarSearch.xml`
Security - Self Registration	Public user registration	`oracle/webcenter/security/view/selfregistration/jsf/taskflows/selfreg-task-flow-definition.xml`
Security - Self Registration - Public Invitation	Public user invitation to join spaces	`oracle/webcenter/security/view/selfregistration/jsf/taskflows/invitation-task-flow-definition.xml`
Security - Enterprise Role - Members	Lists the members of an enterprise group	`oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-task-flow-definition.xml`
Security - Enterprise Role - Members Search	The members of an enterprise group of a particular pattern	`oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-psearch-task-flow-definition.xml`
Security - Enterprise Role - Members Viewer	A tabbed page which shows the members of a group in one and the other tab lets you search for a particular pattern	`oracle/webcenter/security/view/rolemembers/jsf/taskflows/rolemembers-browser-task-flow-definition.xml`
Security - Role Manager	Create, modify and delete application roles	`oracle/webcenter/security/view/rolemanager/jsf/taskflows/global-rolemanager-task-flow-definition.xml`
Security - External Application Credential Provisioning	Credential provisioning for a particular external application	`oracle/adfinternal/extapp/view/fragments/extapp-credential-provisioning-taskflow.xml`
Security - External Application Change Password	Screen to change the password for all the external applications created	`oracle/adfinternal/extapp/view/fragments/extapp-change-password-taskflow.xml`
Similar Items	WebCenter Portal Activity Graph Service View	`view/oracle/webcenter/activitygraph/controller/taskflows/similar-items.xml`
Similar Spaces	WebCenter Portal Activity Graph Service View	`view/oracle/webcenter/activitygraph/controller/taskflows/similar-group-space.xml`
Smart Tag Actions		`oracle/webcenter/collab/smarttag/view/jsf/taskflows/menudata-task-flow-definition.xml`
Smart Tag Actions Enabler		`oracle/webcenter/collab/smarttag/view/jsf/taskflows/action-enabler-task-flow-definition.xml`
Spaces	WebCenter Portal: Spaces View	`oracle/webcenter/community/view/taskflows/browsers/CommunityBrowserRegion.xml`
Space Members	WebCenter Portal: Spaces View	`oracle/webcenter/webcenterapp/view/taskflows/admin/table-of-members-taskflow.xml`
Subscription Preferences	WebCenter Portal Notification Service View	`oracle/webcenter/notification/view/jsf/regions/SubscriptionPreferences.xml`
Subscription Viewer	WebCenter Portal Notification Service View	`oracle/webcenter/notification/view/jsf/regions/SubscriptionsViewer.xml`

Table 25–1 (Cont.) WebCenter Portal Task Flows

Description	Library/JAR Files	Definition Path
Tag Cloud	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/tag-selection.xml`
Tagging Dialog	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/launch-dialog.xml`
Tagged Items	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/related-resources.xml`
Tagging - Personal View	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/tagging-personal-view.xml`
Tagging - Related Links	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/related-links.xml`
Tagging - Similar Items	WebCenter Portal Tagging Service View	`oracle/webcenter/tagging/controller/taskflows/related-links.xml`
taskflow-renderer	WebCenter Portal: Framework	`oracle/webcenter/portal/framework/navigation/renderer/taskflow-renderer.xml`
Worklist	WebCenter Portal Worklist Service View	`oracle/webcenter/worklist/view/jsf/taskFlowDefs/worklist.xml`

Part IV

Integrating and Publishing Content

Part IV contains the following chapters:

- Chapter 26, "Introduction to Integrating and Publishing Content"
- Chapter 27, "Managing Content Repository Connections"
- Chapter 28, "Working with Content Data Controls"
- Chapter 29, "Creating Content Presenter Display Templates"
- Chapter 30, "Integrating the Documents Service"
- Chapter 31, "Adding Content Task Flows and Document Components to a Portal Page"
- Chapter 32, "Integrating Wikis and Blogs"

26

Introduction to Integrating and Publishing Content

There are several ways to integrate content into a WebCenter Portal: Framework application. The method you choose is dependent on the requirements of the application and how you want to expose content to end users.

You can integrate content into a Framework application using any of the following methods (for more information, see the references at the end of this chapter):

- **Content data controls** use JCR adapters to enable read-only access to content in a Content Server, Oracle Portal, or SharePoint content repository, and maintain tight control over the way the content displays in a Framework application.

  This functionality is available primarily for backward compatibility with prior releases, and for requirements outside the capability of Content Presenter or the Documents service and its task flows.

- **Content Management Interoperability Services (CMIS) REST APIs** surface and manage content in Content Server.

- **Content Presenter** enables you to precisely customize the selection and presentation of content in a Framework application. The Content Presenter task flow is available only when the connected content repository is Content Server and your WebCenter Portal administrator has completed the prerequisite configuration. With Content Presenter, you can select a single item of content, contents under a folder, a list of items, query for content, or select content based on the results of a Personalization Conductor scenario, and then select a template to render the content on a page in a Framework application. Content Presenter has no dependency on the Documents service for adding or managing the content it displays.

- **Documents service** task flows and components offer a variety of formats to display folders and files, including wikis and blogs, on a page in a Framework application. You can choose the task flows appropriate for your application to provide features for accessing, adding, and managing folders and files; configuring and viewing file and folder properties; and searching file and folder content in in Content Server, Oracle Portal, or SharePoint content repositories.

> **Note:** The availability of SharePoint as a content repository requires the installation of the SharePoint adapter, as described in Section 3.6, "Installing the Oracle WebCenter Adapter for SharePoint." Administration for SharePoint is performed using WLST commands, not Oracle Enterprise Manager Fusion Middleware Control Console.

Using Documents service task flows and document components (such as links, previews, and images), you can add content to the application, and also provide end users with content and Documents service task flows built into the application to manage, display, and search documents at runtime.

Table 26–1 provides a comparative overview of these methods to help you select the most appropriate method for your needs.

Table 26–1 Methods of Integrating Content into a Framework Application

	Content Data Controls	Content Management REST APIs	Content Presenter	Documents Service
Repository	Content Server Oracle Portal SharePoint	Content Server	Content Server	Content Server Oracle Portal SharePoint
Content Types	Folders and content files.	Folders and content files with support for metadata properties.	Folders and content files with support for metadata properties. Supports Oracle Site Studio region definitions-based content.	Folders and content files. Content Server only: supports folder and content file metadata properties.
Content Display	Surface content using ADF render components: ADF Go Link, ADF Go Button, ADF Image, and ADF inline frame functions.	Surface content using REST APIs and custom client or server side application code. Surface content through single item selection, by folder, and by query results.	Task flow-based component intended primarily for rendering content. Surface content through single item selection, by folder, and by query results. Surface content in display templates: either built-in templates or custom display templates developed in JDeveloper. Reuse Oracle Site Studio display templates for Site Studio content. Supports WebCenter Personalization Services Conductor scenario (query results).	Task flow-based components intended primarily for collaborating and managing content. Content Server only: ■ Document preview ■ View links ■ View images ■ View file information (properties, version history) ■ View metadata

Table 26–1 (Cont.) Methods of Integrating Content into a Framework Application

	Content Data Controls	Content Management REST APIs	Content Presenter	Documents Service
Content Management	None.	Manage content using REST APIs and custom client or server side application code. Create, update, delete folders and content files and associated metadata fields.	In-context contribution editing for HTML and Site Studio content.	Manage content through graphical user interface. Create, update, delete folders and content files. Content Server only: ▪ Manage associated metadata properties ▪ Item-level security ▪ Oracle Workflow (WebCenter Portal: Spaces only) ▪ View metadata
Benefits	Standard JCR API integrates with many different content repositories.	Flexible REST-based APIs useful for client-side style development.	Flexible display using display templates. Oracle Site Studio support.	Choice of task flows to provide easy UI access to managing content.
Limitations	Read-only content. No Oracle Site Studio support.	Content must reside in Oracle Content Server repository. No Oracle Site Studio support.	Content must reside in Content Server repository.	No Oracle Site Studio support.

The following chapters provide information that you will need for *any* method you use:

- "Managing Content Repositories" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* to configure and manage content repositories used by WebCenter Portal applications.
- Chapter 27, "Managing Content Repository Connections" to configure and edit content repository connections that provide access to decentralized content.

The following chapter provides information about using *content data controls*:

- Chapter 28, "Working with Content Data Controls" to use Java Content Repository (JCR) data controls to enable read-only access to any content repository.

The following manual provides information about using *CMIS (Content Management Interoperability Services) REST APIs*:

- *Oracle Fusion Middleware Content Management REST Service Developer's Guide*

The following chapters provide information about using *Content Presenter*:

- Chapter 29, "Creating Content Presenter Display Templates"
- Chapter 31, "Adding Content Task Flows and Document Components to a Portal Page"

- "Publishing Content Using Content Presenter" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* to work with content at runtime in WebCenter Portal applications.

The following chapters provide information about using the *Documents service*, which includes the Documents service task flows, document components (links, inline frames, and images), wikis, and blogs:

- Chapter 30, "Integrating the Documents Service"
- Chapter 31, "Adding Content Task Flows and Document Components to a Portal Page"
- Chapter 32, "Integrating Wikis and Blogs"
- "Working with the Documents Service Task Flows and Document Components" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* to work with content at runtime in WebCenter Portal applications.

27

Managing Content Repository Connections

There are several ways to integrate content into a WebCenter Portal: Framework application. For more information, see Chapter 26, "Introduction to Integrating and Publishing Content."

This chapter discusses how to configure connections to content repositories to provide access to the content, and how to add content to pages in a Framework application.

This chapter includes the following sections:

- Section 27.1, "Overview of Content Adapters"
- Section 27.2, "Configuring Content Repository Connections"
- Section 27.3, "Editing Content Repository Connections"
- Section 27.4, "Using an Existing Repository Connection for a New WebCenter Portal: Framework Application"

27.1 Overview of Content Adapters

In your Framework application, you can use the following adapters:

- Oracle WebCenter Content: Content Server: The Content Server adapter is used to integrate content from an automated information system managed by Content Server. This system enables sharing, managing, and distributing of business information through a Web site as a common access point.

  This adapter is bundled with Oracle WebCenter Portal extension bundle, so it is integrated by default.

- Oracle Portal: The Oracle Portal adapter is used to integrate content from a content repository located in the Oracle Portal schema.

  This adapter is bundled with Oracle WebCenter Portal extension bundle, so it is integrated by default.

- Oracle WebCenter Adapter for SharePoint: This adapter is used to integrate content from the Microsoft SharePoint 2007 repository.

  You must install this adapter, as described in Section 3.6, "Installing the Oracle WebCenter Adapter for SharePoint."

- File System: The File System adapter is used to add content located on your operating system's file system to JSF pages.

  This adapter is bundled with Oracle WebCenter Portal extension bundle, so it is integrated by default.

Note: The File System adapter is intended to be used in the development environment only.

27.2 Configuring Content Repository Connections

Content repository connections are required to access the repository content to be published on JSF pages. These connections are also required to access the repository content through the WebCenter Portal REST API.

This section describes how to configure content repository connections based on Oracle WebCenter Content: Content Server, Oracle Portal, Oracle WebCenter Adapter for SharePoint, and file system adapters. Content Presenter can connect to and retrieve content available in Content Server content repositories. With JCR data controls, you can connect to Content Server, Oracle Portal, Oracle WebCenter Adapter for SharePoint, and file systems.

Connections can be created under Application Resources in the Application Navigator and under IDE Connections in the Resource Palette.

This section includes the following subsections:

- Section 27.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter"
- Section 27.2.2, "How to Create a Content Repository Connection Based on the Oracle Portal Adapter"
- Section 27.2.3, "How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint"
- Section 27.2.4, "How to Create a Content Repository Connection Based on the File System Adapter"
- Section 27.2.5, "What Happens When You Create a Repository Connection"
- Section 27.2.6, "What You May Need to Know When Creating a Repository Connection"

27.2.1 How to Create a Content Repository Connection Based on the Oracle Content Server Adapter

This section describes how to configure a Content Server-based content repository connection. To successfully perform the subsequent steps, you must use Content Server release 11.1.1.4.0 or 10.1.3.5.1. The release 11.1.1.4.0 or 10.1.3.5.1 are required for Content Presenter. For information about how to configure Content Server for Oracle WebCenter Portal, see "Oracle WebCenter Content: Content Server Prerequisites" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

For an overview of the prerequisites and tasks required to get Content Server working in WebCenter Portal applications (Spaces and Framework applications), see the section "Configuration Roadmap for Content Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To create a Content Server-based repository connection:

1. In Oracle JDeveloper, open your Framework application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection**, then **Content Repository**.

3. In the Create Content Repository Connection dialog, specify a connection name for the connection; for example, `MyOCSConnection`.

   Tip: You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **Oracle Content Server** from the **Repository Type** box.

5. Select **Set as primary connection for Documents service**, to make this connection the primary (default) connection for all content management operations through the Documents service or Content Presenter. Selecting this option writes a configuration to the `adf-config.xml` file that identifies this connection as the default connection. If `connectionName` parameter is specified for a Documents service task flow, that value overrides this setting.

   Note: To be able to set a connection as primary (default) connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly, they must be added to applications.

6. Under **Configuration Parameters**, enter values for the parameters, as shown in Table 27–1 and Figure 27–1.

Table 27–1 Configuration Parameters for Content Server

Configuration Parameters	Values
RIDC Socket Type	Determines whether the client library connects on the Content Server listener port or the Web server filter. It accepts `socket`, `socketssl`, `web`, or `jaxws`. `socket`: Uses an `intradoc` socket connection to connect to the Content Server. The client IP address must be added to the list of authorized addresses in the Content Server. This corresponds to the `SocketHostAddressSecurityFilter` setting in the Content Server configuration file. `socketssl`: Uses an `intradoc` socket connection to connect to the Content Server that is secured using the SSL protocol. The client's certificates must be imported in the server's trust store for the connection to be allowed. `web`: Uses an HTTP(S) connection to connect to the Content Server. `JAX-WS` - Uses an HTTP(S) connection to connect to Content Server. Table 27–2 includes more information on the configuration parameters required for each RIDC socket type.
Server Host Name	Host name of the system where the Content Server is running. For example: `mycontentserver.mycompany.com`.
Content Server Listener Port	Port of the server specified in the **Server Host Name** field. This corresponds to the `IntradocServerPort` setting in the Content Server configuration file, which defaults to port `4444`.

Table 27–1 (Cont.) Configuration Parameters for Content Server

Configuration Parameters	Values
URL of the Web Server Plugin	If the RIDC socket type is `web`, then the URL must be: For Content Server release 11.1.1.4.0: `http://host_name:port_number/_dav/web_root/plugin_root`. For example: `http://mycontentserver:4444/_dav/cms/idcplg` For Content Server release 10.1.3.5: `http://host_name:port_number/web_root/plugin_root`. For example: `http://mycontentserver:4444/cs/idcplg`
Web Server Context Root	The Web server context root for Content Server to integrate advanced metadata and Site Studio capabilities in the Framework application. The format of the Web server context root is `/context_root`. For example, `/cs`.
Admin Username	The user name with administrative rights for the Content Server instance. It defaults to `sysadmin`.
Admin Password	The password for the Content Server admin user. For example: `password`. This is needed for `web` connection only.
KeyStore File Location	Location of key store that contains the private key used to sign the security assertions. The key store location must be an absolute path. For example: `c:\keys\keystore.xyz`.
KeyStore Password	The password required to access the keystore.
Private Key Alias	The client private key alias in the keystore. The key is used to sign messages to the server. The public key corresponding to this private key must be imported in the server keystore.
Private Key Password	The client private key password required to retrieve the key from the keystore.
JAX-WS Client Security Policy	Enter the client security policy to be used when the **RIDC Socket Type** is **JAX-WS**. The JAX-WS client security policy can be any valid OWSM policy, but must match the security policy configured for the Content Server's Native Web Services IdcWebLogin service. For more information about the IdcWebLogin service, see "WebCenter Content Web Services" in the *Oracle WebCenter Content Developer's Guide for Content Server*. For information about predefined security policies, see the section "Security Policies" in *Oracle Application Server Web Services Security Guide*. For information about configuring a security policy on client side, see the section "Configuring Clients" in *Oracle WebCenter Content Developer's Guide for Content Server*. For information about configuring a security policy on the server side, see the section "Configuring WS-Security through WS-Policy" in *Oracle WebCenter Content Developer's Guide for Content Server*.
Cache Invalidation Interval	The interval (in minutes) used by WebCenter caches to automatically detect external Content Server content changes. This allows WebCenter to automatically clear cached items when changes to those items are made directly in the Content Server UI. The interval is in minutes. A value of `0` disables cache invalidation. The minimum valid value to enable cache invalidation is `2`.

Table 27–1 (Cont.) Configuration Parameters for Content Server

Configuration Parameters	Values
Binary Cache Maximum Entry Size	The maximum size (in bytes) for WebCenter caching of Content Server binary documents. Documents larger than this size are not cached by WebCenter. The unit is bytes and defaults to `102400` (`100 Kb`).

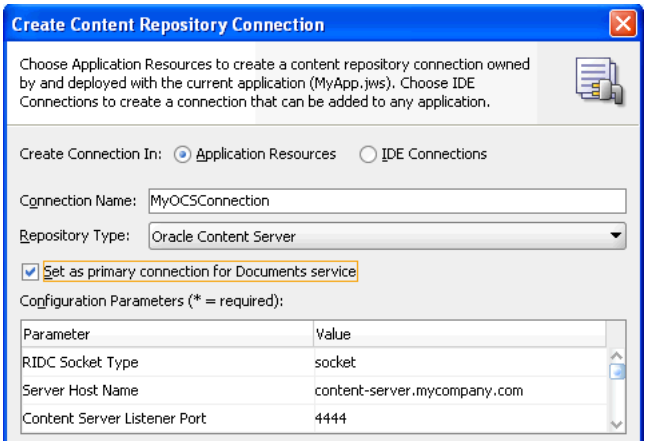
Table 27–2 Content Server Connection Parameters for Each RIDC Socket Type

Connection Parameters	RIDC Socket Type: web	RIDC Socket Type: socket	RIDC Socket Type: socketssl	RIDC Socket Type: jaxws
Server Host Name	Not Applicable	Mandatory for Content Presenter Defaults to local host.	Mandatory for Content Presenter Defaults to local host.	Not Applicable
Content Server Listener Port	Not Applicable	Port specified for the incoming provider in the server. Defaults to `4444`.	Port specified for the `sslincoming` provider in the server. Defaults to `4444`.	Not Applicable
URL of the Web Server Plugin	Mandatory	Not Applicable	Not Applicable	Mandatory Use format: `http://`*`hostname:port`*`/web_root` For example: http(s)://*`myhost.com:9044`*`/idcnativews`
Web Server Context Root	Optional	Optional	Optional	Optional
JAX-WS Client Security Policy	Not Applicable	Not Applicable	Not Applicable	Mandatory, unless Global Policy Attachment is used, in which case it should be left empty. This is the name of an OWSM client security policy. This policy must match the corresponding server side policy applied on the Content Server. For information about security policies, see the section "Security Policies" in *Oracle Application Server Web Services Security Guide*.

Table 27–2 (Cont.) Content Server Connection Parameters for Each RIDC Socket Type

Connection Parameters	RIDC Socket Type: web	RIDC Socket Type: socket	RIDC Socket Type: socketssl	RIDC Socket Type: jaxws
Admin Username	Mandatory	Optional Defaults to `sysadmin`	Optional Defaults to `sysadmin`	Optional Defaults to `weblogic`
Admin Password	Mandatory	Not Applicable	Not Applicable	Whether the password is used or not depends on the selected JAX-WS security policy.
Key Store and Private Key	Not Applicable	Not Applicable	Mandatory	Not Applicable
Identity Propagation	Not supported at runtime	Supported For testing purpose, connects as guest if no/invalid user name is specified.	Supported For testing purpose, connects as guest if no/invalid user name is specified.	Supported, provided that the JAX-WS security policy supports identity propagation.
External Application	Mandatory	Supported Password is not used.	Supported Password is not used.	Supported Whether the password is used or not depends on the selected JAX-WS security policy.
User Name and Password	Supported	Supported The password is not verified during the test connection operation.	Supported The password is not verified during the test connection operation.	Supported Whether the password is used or not depends on the selected JAX-WS security policy.

Figure 27–1 Content Repository Connection - Content Server



7. In the **Login Timeout (ms)** field, specify the time in milliseconds. This timeout determines how long the application must wait when trying to establish a session with the content repository. If the network connection or the content repository

server to which you are trying to connect is slow, then consider overriding the default value specified in this field.

8. Select the applicable authentication method. See Section 27.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods" for information. If you select the Identity Propagation authentication method, then you must configure secure socket layer (SSL) in Content Server. For information, see the section "Oracle WebCenter Content: Content Server - Security Considerations" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

9. **User Name** and **Password** fields under the **Specify login credentials for the current JDeveloper session** checkbox are optional for the Identity Propagation authentication method. In a secured application when Identity Propagation is the chosen authentication method, selecting the Specify login credentials for the current JDeveloper session checkbox lets you navigate to the authenticated content repository connection at design time. When this checkbox is selected, you can also validate if user name and password you specified can be authenticated.

   Note: If the **Specify login credentials for the current JDeveloper session** checkbox is selected, then the credentials you entered are used. If the checkbox is not selected, then the connection is tested using External Application credentials (if they exist), otherwise null credentials are used.

   If you have selected **External Application** for authentication and also specified either public or shared credentials, then you can leave these fields blank, as the public or shared credentials can be used to login at design time. However, if you have selected **External Application** but have not specified public or shared credentials, then you must specify user name and password here.

10. Click **Test**.

    During testing user name and password are used (if specified). If they are not specified, but an external application is defined with either shared or public credentials, then this is used. Otherwise the guest user is used.

11. Click **OK**.

12. In the Application Resources panel, expand the repository connection you just created. The Connection Credentials dialog displays.

    Note: The Connection Credentials dialog displays if you do not save the credentials, that is, you do not select the **Specify login credentials for the current JDeveloper session** checkbox, or if you do not specify an External Application service that uses public or shared credentials.

13. Enter the **User Name** and **Password** for the Content Server connection and click **OK**. The connection displays under the Application Resources panel.

27.2.2 How to Create a Content Repository Connection Based on the Oracle Portal Adapter

This section describes how to create a content repository connection based on the Oracle Portal adapter. Before creating a repository connection, see Section 27.2.6, "What You May Need to Know When Creating a Repository Connection". You can use this connection to configure a content data control that will enable you to add content from the Oracle Portal repository to JSF pages.

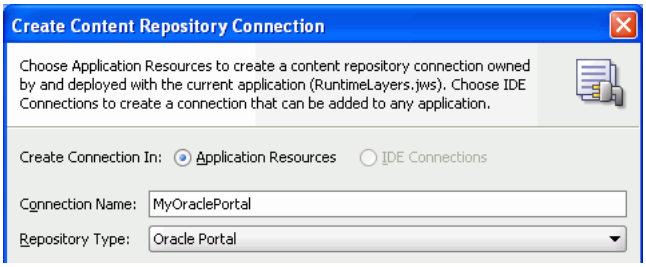
To create an Oracle Portal repository connection:

1. In Oracle JDeveloper, open your Framework application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection** and then **Content Repository**.
3. In the Create Content Repository Connection dialog, specify a name for the connection; for example, `MyOraclePortal`.

   Tip: You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **Oracle Portal** from the **Repository Type** box, as shown in Figure 27–2.

Figure 27–2 Content Repository Connection - Oracle Portal



5. Select **Set as primary connection for Documents service** to make this connection the primary (default) connection for all content management operations through the Documents service or Content Presenter. Selecting this option writes a configuration to the `adf-config.xml` file that identifies this connection as the default connection. If `connectionName` parameter is specified for a Documents service task flow, that value overrides this setting.

   Note: To be able to set a connection as primary (default) connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly, they must be added to applications.

6. Select a connection from the **Database Connection** list, or create a new connection to the Oracle Portal schema:
   a. In the Create Database Connection wizard, specify the connection name and type.
   b. Enter the database user name and password of the Oracle Portal schema. By default, the user is PORTAL.

c. Under **Oracle (JDBC) Settings**, if you intend to specify a custom JDBC URL, then select the **Enter Custom JDBC URL** checkbox, and specify the database URL of the Portal schema in the `jdbc` format:

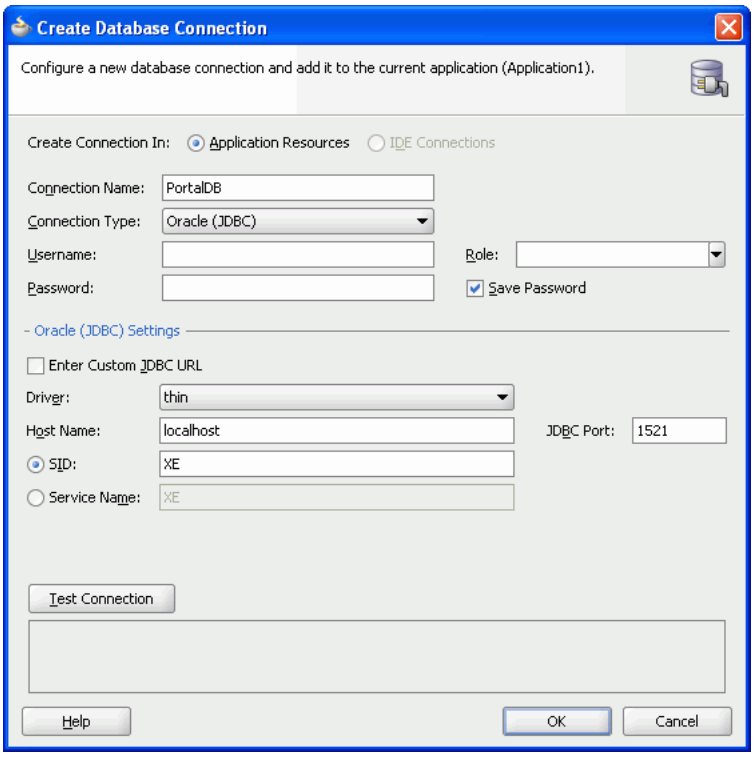
```
jdbc:oracle:thin:@dbhost:dbport:dbsid
```

d. If you did not perform the above step, then select a driver, enter the host name, JDBC port, SID, and service name, as shown Table 27–3 and Figure 27–3.

Table 27–3 Parameters for Creating Oracle Portal-based Content Data Control

Parameters	Description
Driver	There are two types of drivers: thin and oci8. The thin driver can be used to connect to Oracle Database release 8*i* or later databases with TCP/IP network protocols. This driver is included in the default Oracle JDBC library for all projects. The oci8 driver is used when creating a Java application that runs against an Oracle Application Server. This is a thick driver optimized for the Oracle Database. It is a mix of Java and native code. This driver handles any database protocol (TCP, IPX, BEQ, and so on). It is recommended for applications that are run from the computer on which they are stored.
Host Name	Name of the Oracle Database. Use an IP address or a host name that can be resolved by TCP/IP; for example, `myserver`. The default value is `localhost`.
JDBC Port	Value to identify the TCP/IP port.
SID	Unique system identifier of an Oracle database instance.
Service Name	The service name for an Oracle database instance.

Figure 27–3 Create Database Connection



e. Click **OK**.

7. In the Create Content Repository Connection dialog, in the **Login Timeout (ms)** field, specify the time in milliseconds. This timeout determines how long the application must wait when trying to establish a session with the content repository. If the network connection or the content repository server to which you are trying to connect is slow, then consider overriding the default value specified in this field.

8. Select the applicable authentication method. See Section 27.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods" for information.

9. Select the **Specify login credentials for current JDeveloper session** checkbox and enter the Oracle Portal single sign-on user name and password for logging in to your portal instance. The Oracle Portal single sign-on user name and password entered here are used to test the connection and are functional only at design time. Then click **OK**.

Note: If the **Specify login credentials for current JDeveloper session** checkbox is selected, then the credentials you entered are used. If the checkbox is not selected, then the connection is tested using External Application credentials (if they exist), otherwise null credentials are used.

If you have selected **External Application** for authentication and also specified either public or shared credentials, then you can leave these fields blank, as the public or shared credentials can be used to login at design time. If you specified both public and shared credentials for the external application, then the public credentials have higher precedence. However, if you have selected **External Application** but have not specified public or shared credentials, then you must specify user name and password here.

10. In the Application Resources panel, expand the repository connection you just created. The Connection Credentials dialog displays.

Note: The Connection Credentials dialog displays if you do not save the credentials, that is, you do not select the **Specify login credentials for the current JDeveloper session** checkbox, or if you do not specify an External Application service that uses public or shared credentials.

11. Enter the user name and password for Oracle Portal connection and click **OK**. The connection displays under the Application Resources panel.

See Also: To create a data control using this connection, perform the procedure described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".

27.2.3 How to Create a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint

The Oracle WebCenter Adapter for SharePoint extracts and searches content within a Microsoft SharePoint 2007 repository. The adapter accesses the repository using the Microsoft SharePoint SOAP interfaces. Oracle recommends that you study Section 27.2.6.3, "What You Should Know About Oracle WebCenter Adapter for SharePoint." before configuring and using Oracle WebCenter Adapter for SharePoint.

This section includes the following subsections:

- Section 27.2.3.1, "Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint"
- Section 27.2.3.2, "Mapping Microsoft SharePoint Content and Services"

27.2.3.1 Creating a Content Repository Connection Based on the Oracle WebCenter Adapter for SharePoint

To create a repository connection based on Oracle WebCenter Adapter for SharePoint:

1. In Oracle JDeveloper, open your Framework application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection**, then **Content Repository**.

3. In the Create Content Repository Connection dialog, specify a connection name for the connection; for example, `MySPConnection`.

   Tip: You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **JCR SharePoint Adapter** from the **Repository Type** box.

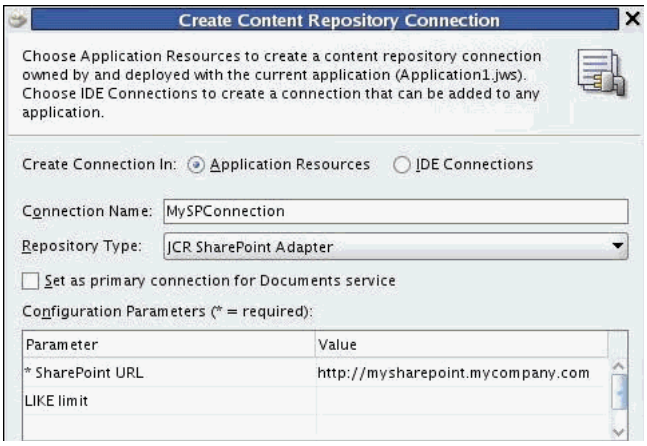
   Note: This option displays only if the extension for Oracle WebCenter Adapter for SharePoint is installed.

5. Select **Set as primary connection for Documents service**, if you intend to make it the default connection for the Documents service. Selecting this option writes a configuration to the `adf-config.xml` file that identifies this connection as the default connection for the Documents service. If a Document Library task flow is used without any `connectionName` input parameter, then this connection is used. For information about the Documents service, see Chapter 30, "Integrating the Documents Service."

   Note: To be able to set a connection as primary connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly, they must be added to applications.

6. Under **Configuration Parameters**, enter the **SharePoint URL** which is the web address of the SharePoint site to which you want to connect. For example, `http://mysharepoint.mycompany.com`, as shown in Figure 27–4.

Figure 27–4 Content Repository Connection - Oracle WebCenter Adapter for SharePoint



7. Enter an integer value representing the number of characters for the LIKE limit. If no value is set, the LIKE limit defaults to `64` characters as the limit for search pattern strings that are passed to the Microsoft SharePoint Server. If the value is set to zero, then the full string is passed to the server. That is, the LIKE limit is disabled. This parameter is optional. Oracle recommends that it be left to its

default state. For related information, see Section 27.2.6.3, "What You Should Know About Oracle WebCenter Adapter for SharePoint."

8. In the **Login Timeout (ms)** field, specify the time in milliseconds. This timeout determines how long the application must wait when trying to establish a session with the content repository. Oracle recommends that there is a fast network connection between the deployed application and the Microsoft SharePoint repository. If the network connection or the content repository server to which you are trying to connect is slow, then consider overriding the default value specified in this field.
9. Select the **External Application** authentication method. For information about this authentication method, see Section 27.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods."

> **Note:** Oracle WebCenter adapter for Microsoft SharePoint does not support the **Identity Propagation** authentication method. Therefore, the **External Application** authentication method lets you use a single shared set of credentials for all users when accessing the repository. You can also configure this option to enable each user to enter their own login credentials when accessing the repository the first time by omitting both the Shared and Public credentials in the External Application configuration. The system then reuses each unique credential on subsequent access requests.

10. Select the **Specify login credentials for the current JDeveloper session** checkbox and specify **User Name** and **Password**. This step is optional, and only required if you want to override the **External Application** authentication credentials created in the previous step. Moreover, this override only applies to the JDeveloper design time.
11. Click **OK**.
12. Optionally, to test the connection, expand the repository connection you just created, in the Application Resources panel. The Connection Credentials dialog displays.

> **Note:** The Connection Credentials dialog displays if you do not save the credentials, that is, you do not select the **Specify login credentials for the current JDeveloper session** checkbox, or if you do not specify an External Application service that uses public or shared credentials.

13. Enter the **User Name** and **Password** for this connection and click **OK**. The connection displays under the Application Resources panel.

    This step is optional.

27.2.3.2 Mapping Microsoft SharePoint Content and Services

The Oracle WebCenter Adapter for SharePoint is designed to map the content managed in SharePoint servers, such as sites, lists, list items, metadata, documents, as well as content services delivered by Microsoft SharePoint, such as search and security.

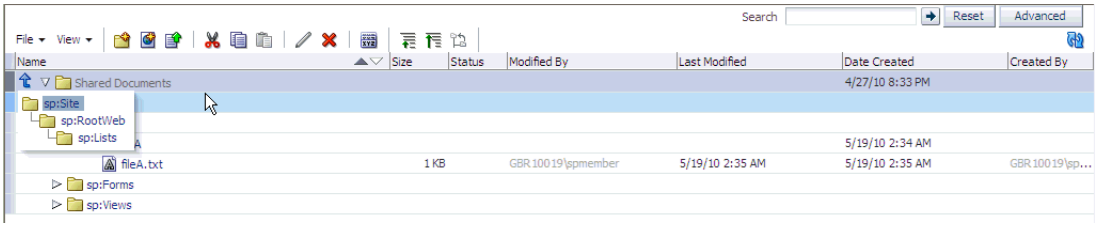
The adapter maps content and content services that are relevant to the JCR standard. In other words, the adapter does not map the Microsoft SharePoint graphical user interface (GUI) and the application-level functions that are not part of the JCR

standard. For example, the adapter does not map the GUI application logic of the Microsoft SharePoint server (not covered by JCR), but it provides GUI-relevant metadata so that JCR or Java applications can restore some of the GUI logic such as SharePoint lists, if that is the goal of the application.

Default JCR Location of the Microsoft SharePoint Documents Libraries

For a SharePoint Services v3 Team Site, the Shared Documents Library appears at path: `/sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files`, as shown in Figure 27–5.

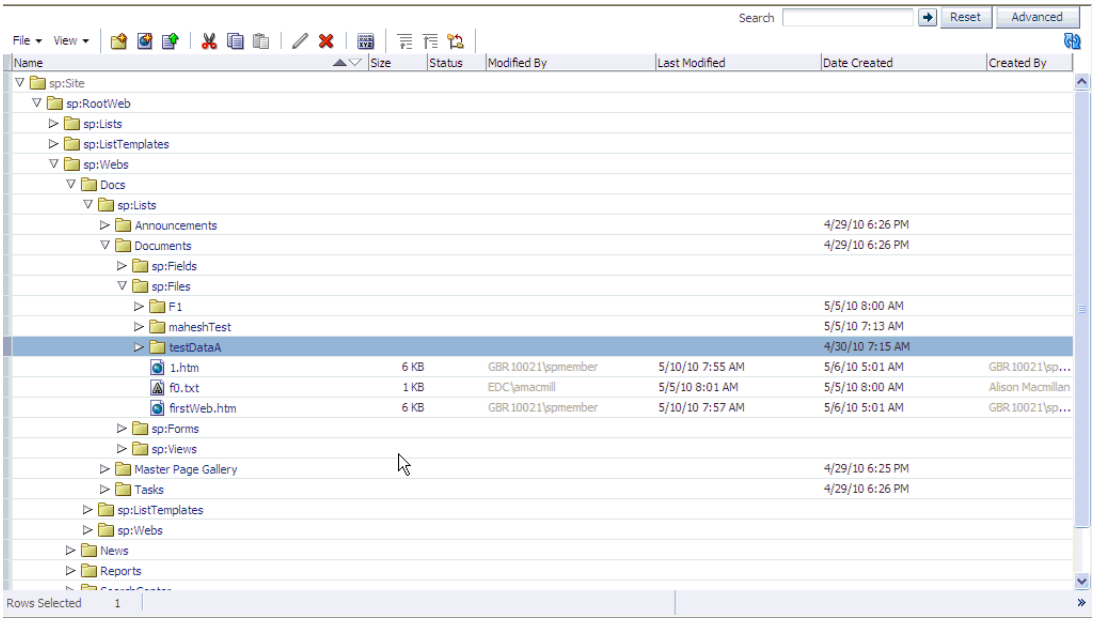
Figure 27–5 The Shared Documents Library Path for a SharePoint Services v3 Team Site



For the MOSS 2007 site created during installation, the Documents Center Document Library appears at path: `/sp:Site/sp:RootWeb/sp:Webs/Docs/sp:Lists/Documents/sp:Files`, as shown in Figure 27–6. This path reflects the structure where the Documents Center is a subsite in the SharePoint main site.

Files are JCR nodes of primary type `nt:file`, and of mix-in type `sp:File`. Folders are JCR nodes of primary type `nt:folder` and of mix-in type `sp:Folder`. The `sp:` mix-ins are similar to `nt:unstructured` in their definitions.

Figure 27–6 The Documents Center Document Library Path for MOSS 2007 Site



27.2.4 How to Create a Content Repository Connection Based on the File System Adapter

This section describes the procedure to create a content repository connection based on the file system adapter. The File System adapter is intended to be used in the development environment only.

To create a File System repository connection:

1. In Oracle JDeveloper, open your Framework application.
2. In the Application Resources panel, right-click **Connections** and choose **New Connection** and then **Content Repository**.
3. In the Create Content Repository Connection dialog, enter a name for the connection; for example, `MyConnection`.

   Tip: You can choose between creating application-specific connections in the Application Resources panel and creating common connections in the Resource Palette (IDE Connections).

4. Select **File System** from the **Repository Type** box.
5. Select **Set as primary connection for Documents service** to make this connection the primary (default) connection for all content management operations through the Documents service or Content Presenter. Selecting this option writes a configuration to the `adf-config.xml` file that identifies this connection as the default connection. If `connectionName` parameter is specified for a Documents service task flow, that value overrides this setting.

   Note: To be able to set a connection as primary (default) connection for the Documents service, you must select the **Application Resources** option in **Create Connection In**. This is because connections created as IDE connections cannot be used directly, they must be added to applications.

6. Under **Configuration Parameters**, select the **Base Path** row, and enter the path to the folder in which your content is placed, for example, `C:\MyContent`.

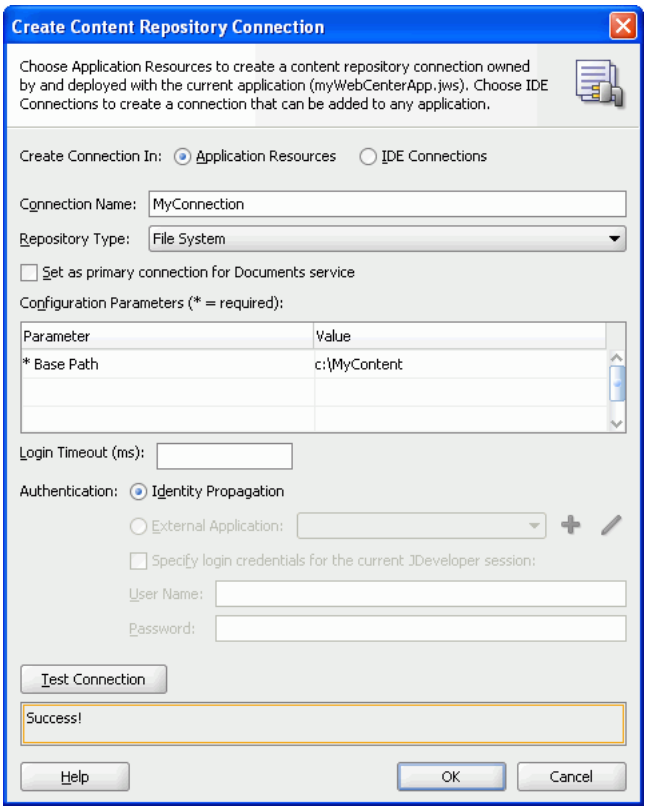
   Note: The following must be considered while creating a file system connection:

   - The base path value is used as the root (`"/"`) for the file system-based data control in Section 28.5, "Integrating Content Using Content Data Controls".
   - When you work on a UNIX system, the File System adapter inherits the case-sensitive file name characteristic of UNIX systems. So, on UNIX systems, you must ensure that references to files follow the same case as that used in the original file names. For example, suppose the `Test.html` file was created on a Microsoft Windows system. When you reference this file on a Linux system, you must ensure that you use `Test.html`, and not `test.html` or `TEST.html`.

7. Leave the **Login Timeout (ms)** field blank.

8. Click **Test Connection** to check whether you have entered the connection details correctly. You should see a `Success!` status, as shown in Figure 27–7.

Figure 27–7 File System Connection



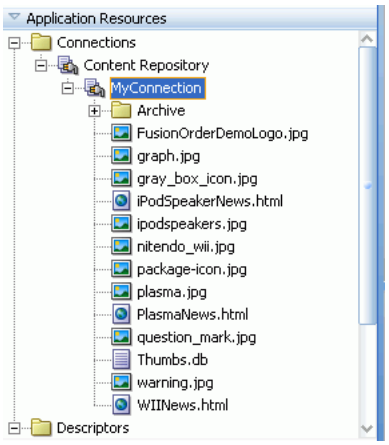
9. Click **OK**.

10. In the Application Resources panel, expand the repository connection you just created.

27.2.5 What Happens When You Create a Repository Connection

When you create a connection to a repository, the contents in the main directory of the repository display under the Content Repository connection in the Application Resources panel, as shown in Figure 27–8. You can double-click folders and files to view them.

Figure 27–8 Application Resources



You can use repository connections to create JCR data controls that enable integration of the repository content with JSF pages. See Section 28.2, "Configuring Content Data Controls for JCR Adapters" and Section 28.5, "Integrating Content Using Content Data Controls" for information. These connections can also be consumed through the Documents service and Content Presenter task flows. See Chapter 30, "Integrating the Documents Service" for information.

27.2.6 What You May Need to Know When Creating a Repository Connection

This section includes:

- Section 27.2.6.1, "What You Should Know About Using Identity Propagation and External Application Authentication Methods"
- Section 27.2.6.2, "What You Should Know About Oracle Portal"
- Section 27.2.6.3, "What You Should Know About Oracle WebCenter Adapter for SharePoint"

27.2.6.1 What You Should Know About Using Identity Propagation and External Application Authentication Methods

The Create Content Repository Connection dialog provides the following options for authentication methods:

- **Identity Propagation**: If you select this option, no credentials are passed to the repository. The repository connector instead uses the current user's identity as determined from the Java security context. This must only be used when the application and the repository use the same identity store to authenticate users.

  To apply this authentication method, it is best to configure security for your application using the Configure ADF Security wizard since repositories may support only authenticated users or provide only limited access to the public or guest user. See Section 28.4, "Securing a Content Repository Data Control".

- **External Application**: The External Application method can be used in all other cases where the current user identity should not be propagated directly to the repository. For more information, see Section 69.13, "Working with External Applications" and Section 30.3, "Setting Security for the Documents Service."

  The External Application service allows different types of credentials to be associated with a connection:

- Public credentials are used whenever an application is not secured, or the user has not yet logged in.
- Shared credentials are used for any authenticated user.
- If shared credentials are not specified, then mapped credentials can be used to obtain a result similar to identity propagation when the application and content repository do not use the same user store. In that case the external application service provides runtime screens through which users can provide their credentials for accessing the content repository. Those credentials are securely stored and used for any future connection that this user tries to establish.

If you intend to configure the External Application service, then click **New** to launch the Register External Application wizard and do the following:

a. Specify name of the external application.

b. In the **Login URL** field, enter the URL of the external application; for example, `http://content-server.mycompany.com/idc/`

> **Note:** This URL is optional and is required only to provide *click through login* to the content repository's own user interface. See the External Application service documentation for more information on how to configure click through login.

c. Then, under **Authentication Details**, select **Basic** from the list. Leave the fields under **Login Details** blank.

This option is selected when *click through login* is not required. If *click through login* is enabled, then you must select another option based on the authentication method used in the repository.

d. Accept the default values and click **Next**.

e. Enter shared credentials that can be used at design time and runtime. Do not specify shared credentials, if users must specify their own credentials.

f. Enter public credentials that can be used at design time and runtime. Specify public credentials only if you intend public users to view documents or contents.

Click **Finish**.

27.2.6.2 What You Should Know About Oracle Portal

The following should be considered while using the Oracle Portal adapter:

- To use Oracle Portal-based content data control capabilities, you can install Oracle Portal release 10.1.4.1, 10.1.4.2, or 11. If you use the Oracle Portal release 10.1.4.1, then you must update it with the required patches. Consult Oracle Application Server Release Notes for Microsoft Windows (for the required platform) for release 11.1.1.0.0 to know the exact patch number.
- Only `file`, `image`, `imagemap`, and `text` item types and custom types based on these item types are supported.
- The `portal:container` page type and its extensions are supported.

- Content is always exposed in the default language of the page group. For example, if there are three page groups with different default languages, then the content displays only for those default languages and not for any translations that exist.

27.2.6.3 What You Should Know About Oracle WebCenter Adapter for SharePoint

Consider the following while configuring and using Oracle WebCenter Adapter for SharePoint:

- The adapter does not support connecting to a Microsoft SharePoint repository configured to allow `anonymous` access.
- There should be a fast network connection between the application server instance on which the adapter is running and the Microsoft SharePoint repository.
- **Microsoft SharePoint versions**: Oracle WebCenter Portal supports the following Microsoft SharePoint versions:
  - Microsoft Office SharePoint Server (MOSS) 2007 SP2
  - Microsoft Windows SharePoint Services (WSS) version 3 SP2
- **Versioning Settings**: Oracle WebCenter Portal supports the following Microsoft SharePoint 2007 Document Library versioning settings:
  - Require Check Out : No
  - Content Approval : No
  - Document Version History : No versioning

  The adapter does not function correctly if any other Versioning Settings are configured. For example, upload operations fails if `Require CheckOut` is set to `yes`. If document version history or content approval are enabled, new versions or documents have restricted visibility.
- **Session-Based Memory Cache**: For performance reasons, Oracle WebCenter Adapter for SharePoint maintains a session-based memory cache. A Framework application, which uses this adapter, must configure an `Http Session time-out` in the application's `web.xml` file to ensure that any unused sessions are closed, and the adapter memory cache is released.
- **Mapping**: The mapping takes into account only those content artifacts and services, which are relevant and defined in the JCR API. Additionally, the adapter maps the content and services to the extent supported by native SharePoint web services, as the adapter does not rely on custom additional server-side components. Consequently:
  - Full text search only returns items that are indexed by the SharePoint server.
  - Set of searchable columns is limited.
  - A fixed set of node types is supported in searches.
  - Some searches require additional network round trip as the adapter does not receive the full information necessary to map results to JCR from the web services.

  The adapter exposes content from a Microsoft SharePoint server in a hierarchical way which is similar to the way by which the content is exposed by the native web interface. The items of the hierarchy have a primary type of either `nt:folder`, `nt:file`, or a specialization of these. Additional SharePoint specific characteristics are in some instances added by means of mixins, that is, `sp:Folder`, `sp:File`, and so on. The SharePoint specific name space, which is

identified by the `sp` prefix in these examples, is used for items controlled by the SharePoint server. Items controlled by the user - webs, documents, list items and the like - are in the default name space. The adapter allows write access to SharePoint document libraries. That is, it allows folders and files to be created within document libraries in correspondence with creating folders or uploading files to the native SharePoint web interface. Examples which demonstrate this hierarchical mapping are described in the next point.

- **Site Structure**: The structure mapping of a SharePoint site is as follows:
  - `/sp:Site`: The root element of the site.
  - `/sp:Site/sp:RootWeb`: The root web element.
  - `/sp:Site/sp:RootWeb/sp:Lists`: The path to the root of all lists in the site.
  - `/sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files`: The path to the start of the file and folder hierarchy within a document library. In this example, the Shared Documents library is shown. All site document libraries are mapped under `/sp:Site/sp:RootWeb/sp:Lists`, each mapped by their own document library name.
  - `/sp:Site/sp:RootWeb/sp:Lists/Shared Documents/sp:Files/myFiles`: An example of a folder within the sites Shared Documents document library.
  - `/sp:Site/sp:RootWeb/sp:Webs/mySubSite`: A site's subsites are all mapped under `/sp:Site/sp:RootWeb/sp:Webs`. This example shows the path to a subsite named `mySubSite`. The subsite may itself contain Document Libraries.
  - The adapter does not support the `write` access to a SharePoint Document Library with settings that require one or more mandatory SharePoint columns.
- **Search and indexing**:
  - **JCR and native Microsoft SharePoint search**: The Oracle WebCenter Adapter for SharePoint uses the native SharePoint search web service to evaluate JCR queries. Features present in JCR 1.0 but not available in the SharePoint search service are usually ignored.
  - **Indexed items**: The native search finds only those items that are indexed by the Microsoft SharePoint server. Other items that may match the query criteria are ignored. By default, SharePoint Services 2007 is configured for search and full-text indexing of a limited set of document types. For more information, and information on how to enable search and indexing in more complex deployments, see Microsoft documentation.
  - **LIKE limit**: The SharePoint native query language uses a LIKE keyword that the adapter uses to constrain queries by URLs (document paths) that match a pattern. The native `LIKE` operator supports a pattern match on strings up to 64 characters. Therefore, the Oracle WebCenter Adapter for SharePoint applies a client-side filtering on result sets to ensure that the correct constraint on URL is applied. The adapter's LIKE limit parameter controls this feature, and if this limit is not set, it defaults to applying a 64 character limit. The parameter can be set to `0` to disable client-side filtering. The parameter can also be set to some other positive value to apply a different character limit. However, this limit can only be increased if the SharePoint instance supports LIKE tests on URLs greater than 64 characters.

- **Creation and Modification timestamps**: When displaying the contents of a SharePoint Document Library, the adapter reports the Creation and Modification timestamps as the times that the document was created on or last updated in the library. In the case of some document types, for example Microsoft Office documents, the native SharePoint search evaluates against the document metadata stored within each document. This can be different from the times when the document was uploaded or updated in the Document Library, and therefore, it appears to give misleading search results.
- **Combining search keywords**: The adapter does not support combining search keywords in an `OR` expression.
- **Search by Last Modifier**: The adapter does not support search by Last Modifier.
- **Supported JCR search operators and properties**: The adapter supports the following:
  * Relational operators for comparison: `=`, `>=`, `>`, `<=`, `<`, `LIKE`, `<>`, `IS NULL`, `IS NOT NULL`. Predicates, descendants, and self operators (`//`) are supported on the last location step.
  * Properties for comparison: Size, Author, Modified, URL, Title, `IsDocument`, `ContentClass`, `SiteName`, Description, `FileName`, `jcr:data`, `jcr:mimeType`, `jcr:created`, `jcr:lastModified`.

27.3 Editing Content Repository Connections

This section describes how you can edit a common or an application-specific content repository connection. This section includes the following subsections:

- Section 27.3.1, "How to Edit a Common Repository Connection"
- Section 27.3.2, "How to Edit a WebCenter Portal: Framework Application-Specific Content Repository Connection"

27.3.1 How to Edit a Common Repository Connection

Common repository connections (IDE connections) are created and edited under the Resource Palette.

To edit a common repository connection:

1. Under the Resource Palette, right-click the repository connection you intend to edit and choose **Properties**, as shown in Figure 27–9. The Edit Repository Connection dialog displays, as shown in Figure 27–10.

Figure 27–9 Resource Palette - Properties

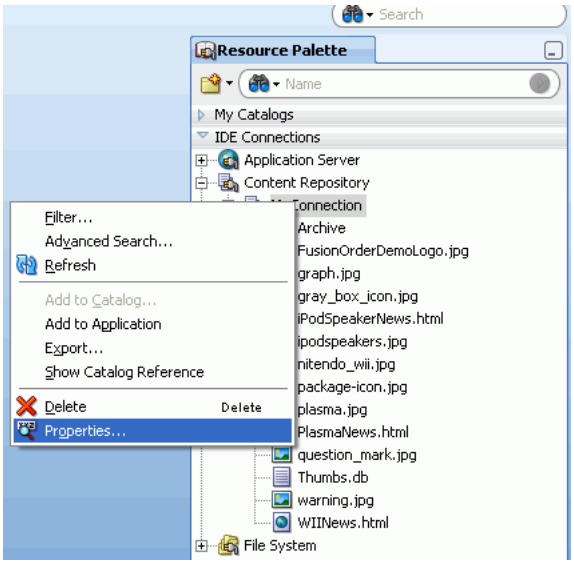
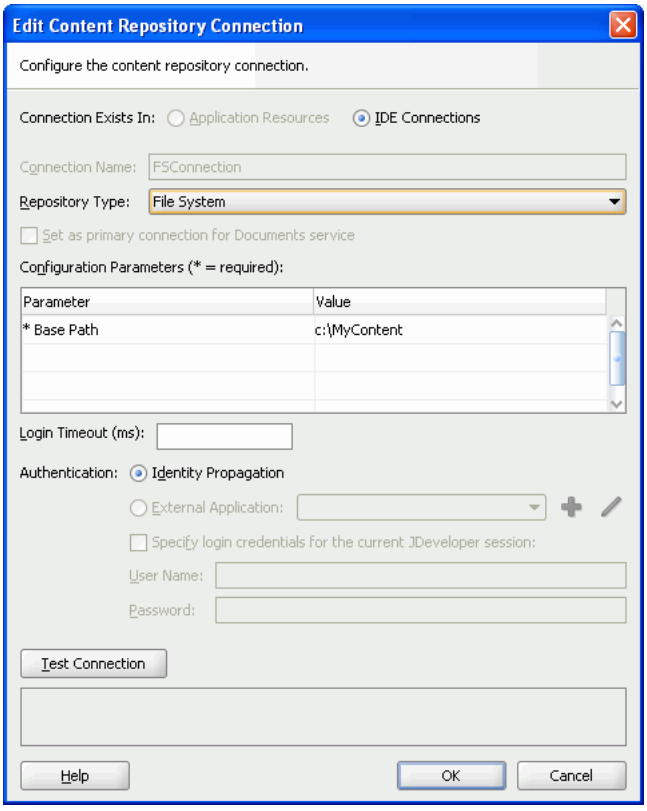


Figure 27–10 Edit Content Repository Connection



2. Change the appropriate parameters.
3. Test the connection and click **OK**.

27.3.2 How to Edit a WebCenter Portal: Framework Application-Specific Content Repository Connection

Application-specific content repository connections exist under the Application Resources panel.

To edit a Framework application-specific content repository connection:

1. In the Application Resources panel, right-click the connection you intend to edit, and choose **Properties**, as shown in Figure 27–11. The Edit Content Repository Connection dialog displays, as shown in Figure 27–12.

Figure 27–11 Application Resources - Properties

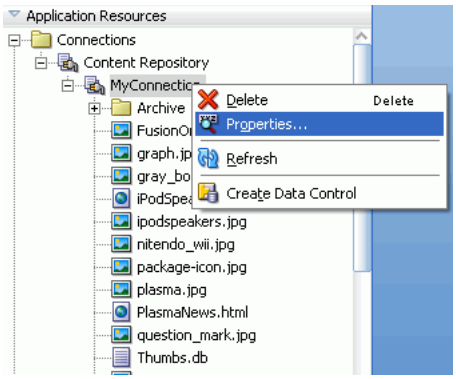
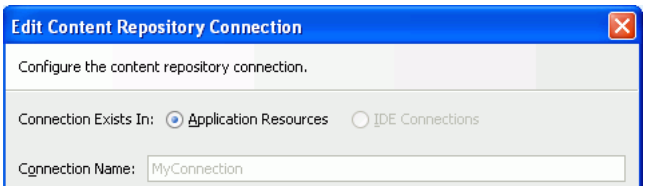


Figure 27–12 Edit Content Repository Connection



2. Change the appropriate parameters. Depending on the repository type, see the earlier sections that describe how to create repository connections. For example, for Content Server, see Section 27.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter".
3. Test the connection and click **OK**.

27.4 Using an Existing Repository Connection for a New WebCenter Portal: Framework Application

You can use an existing repository connection for any Framework application, if you created it as a common repository under the Resource Palette.

To use an existing repository connection:

1. In Oracle JDeveloper, open the Framework application for which you intend to use an existing repository connection.
2. Go to the Resource Palette and select the repository connection that you intend to use for a new application, for example `MyConnection_2`, and drop it under the Application Resources panel of the new application, as shown in Figure 27–13.

Tip: Alternatively, right-click the connection and choose **Add to Application**. The connection is displayed under the Application Resources panel.

Figure 27–13 Dragging an Existing Common Repository Connection (IDE Connection) to an Application



You can now configure a data control for your new application from this repository connection, as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".

28

Working with Content Data Controls

There are several ways to integrate content into a WebCenter Portal: Framework application. For more information, see Chapter 26, "Introduction to Integrating and Publishing Content."

JCR APIs enable independent access to content, regardless of the underlying repository or the type of the content; for example, documents, relational content, and so on. The JCR 1.0 API, as defined in JSR 170, provides a set of basic capabilities for reading, writing, browsing, and searching content. The JCR data controls also enable you to connect to and read from other JCR 1.0 repositories. To add content available in Oracle WebCenter Content: Content Server, Oracle Portal, Oracle WebCenter Adapter for SharePoint, and file systems to JSF pages, you first create connections to the repositories, then use the connections to create data controls based on the repositories.

This chapter includes the following sections:

- Section 28.1, "Overview of Content Data Controls"
- Section 28.2, "Configuring Content Data Controls for JCR Adapters"
- Section 28.3, "Editing Content Repository Data Controls"
- Section 28.4, "Securing a Content Repository Data Control"
- Section 28.5, "Integrating Content Using Content Data Controls"

28.1 Overview of Content Data Controls

A content data control is a container for all the data objects, collections, methods, and operations used to create user interface (UI) components within your Framework application. The data controls provide you with easy-to-use methods that you can drag and drop onto JSF pages to publish content as ADF components, such as URLs, files, and folders. While methods, parameters, and default attributes to publish content are similar across all JCR data controls, the content integration module gives you the flexibility to customize attributes based on your requirements.

Each type of content data control contains methods and parameters to publish content as links, tables, files, and folders, and to add search and advanced search capabilities for your content. Parameters include two types of attributes: default and custom. The default attributes are common across data controls based on File System, Oracle Portal, Oracle WebCenter Content: Content Server, and Oracle WebCenter Adapter for SharePoint. The custom attributes are repository-specific and can be added while creating content data controls.

The following sections describe the methods, parameters, and default attributes that are common across data controls based on File System, Oracle Portal, Oracle WebCenter Content: Content Server, and Oracle WebCenter Adapter for SharePoint:

- Section 28.1.1, "The getItems Method"
- Section 28.1.2, "The search Method"
- Section 28.1.3, "The advancedSearch Method"
- Section 28.1.4, "The getURI Method"
- Section 28.1.5, "The getAttributes Method"

28.1.1 The getItems Method

The `getItems` method returns the files and folders stored starting at a particular location in the repository. This method enables you to publish content in forms, tables, and hierarchical trees. Using this method, you can also create navigation lists and buttons.

Table 28–1 describes the parameters of the `getItems` method.

Table 28–1 Parameters of the getItems Method

Parameter	Description
`path`	Defines the starting point for `getItems`.
`type`	Specifies what should be returned: only files, only folders, or any object.

The `getItems` method returns the attributes described in Table 28–2.

Table 28–2 Return Attributes of the getItems Method

Parameter	Description
`icon16`	Provides a URI to the icon that the Documents service uses for an item.
`icon32`	Provides a URI to the icon that the Documents service uses for an item.
`ID`	Parameter to the Documents service task flows that require an ID.
`lastModified`	Describes the last modified date of an item.
`name`	Describes the name of the returned file or folder.
`path`	Describes the location of the returned file or folder within the content repository.
`URI`	The direct access URL of a file or folder.
`primaryType`	Describes whether the returned object is a file or folder or some other type.

28.1.2 The search Method

The `search` method enables you to create a simple search by name pattern or keyword.

Table 28–3 describes the parameters of the `search` method.

Table 28–3 Parameters of the search Method

Parameter	Description
`path`	Starting path of the search.

Table 28–3 (Cont.) Parameters of the search Method

Parameter	Description
isRecursive	Specifies whether only the specified folder (=false) or the whole tree starting at the specified path (=true) should be searched. The default value is false, that is, if this field is left blank, then the search is performed in the specified folder. Only if the value is true, the search is performed in the entire hierarchy.
keyword	Search keyword for full text search.
namePattern	Pattern search on name. Use the % wildcard to match any number of characters and the _ wildcard to match one character.

The search method returns the attributes described in Table 28–4.

Table 28–4 Return Attributes of the search Method

Parameter	Description
icon16	Provides a URI to the icon that the Documents service uses for an item.
icon32	Provides a URI to the icon that the Documents service uses for an item.
ID	Parameter to the Documents service task flows that require an ID.
lastModified	Describes the last modified date of an item.
name	Describes the name of the returned file or folder.
path	Describes the location of the returned file or folder within the content repository.
URI	The direct access URL of a file or folder.
primaryType	Describes whether the returned object is a file or folder or some other type.

28.1.3 The advancedSearch Method

The advancedSearch method enables you to perform an advanced search by creating a set of search criteria out of any available attribute.

Table 28–5 describes the parameters of the advancedSearch method.

Table 28–5 Parameters of the advancedSearch Method

Parameter	Description
path	Starting path of the search.
isRecursive	Specifies whether only the specified folder (=false) or the whole tree starting at the specified path (=true) should be searched. The default value is false, that is, if this field is left blank, then the search is performed in the specified folder. Only if the value is true, the search is performed in the entire hierarchy.
keyword	Search keyword for full text search.
namePattern	Pattern search on name. Use the % wildcard to match any number of characters and the _ wildcard to match one character.
matchAny	Specifies whether all predicates (=false) or any predicate (=true) should be matched.

Table 28–5 (Cont.) Parameters of the advancedSearch Method

Parameter	Description
`predicates`	A collection of `SimplePredicate` parameters that consist of attributes, comparators, and values.
`type`	Specifies what should be returned: only files, only folders, or any object.

The `advancedSearch` method returns the attributes described in Table 28–6.

Table 28–6 Return Attributes of the advancedSearch Method

Parameter	Description
`icon16`	Provides a URI to the icon that the Documents service uses for an item.
`icon32`	Provides a URI to the icon that the Documents service uses for an item.
`ID`	Parameter to the Documents service task flows that require an ID.
`lastModified`	Describes the last modified date of an item.
`name`	Describes the name of the returned file or folder.
`path`	Describes the location of the returned file or folder within the content repository.
`URI`	The direct access URL of a file or folder.
`primaryType`	Describes whether the returned object is a file or folder or some other type.

28.1.4 The getURI Method

The `getURI` method returns the URI attribute, which is the direct access URL of the file or folder. Its `path` parameter describes the path to the object. You can use this method to create links to content and to inline content in your page. The `getURI` method returns the `URI` attribute.

28.1.5 The getAttributes Method

The `getAttributes` method returns the list of attributes and their values for a given file or folder. Its `path` parameter describes the path to the object.

Table 28–7 describes the attributes that the `getAttributes` method returns.

Table 28–7 Return Attributes of the getAttributes Method

Parameter	Description
`name`	Name of the attribute.
`value`	Value of the attribute.

28.2 Configuring Content Data Controls for JCR Adapters

This section describes how to create a content data control based on a content repository connection:

- Section 28.2.1, "How to Configure a Content Repository Data Control"

- Section 28.2.2, "What Happens When You Configure a Content Repository Data Control"

28.2.1 How to Configure a Content Repository Data Control

The procedure to create content repository data controls is the same irrespective of the types of the connections that you use to create data controls. If you have not created a connection to your content repository, then see Section 27.2, "Configuring Content Repository Connections". In this section, a content data control is created using a Content Server-based connection.

To create a content data control using an existing content repository connection:

1. In the Application Navigator, select the **Portal** project, in which the project entries for the new data control should be created.

   You must do this to ensure that the data control definition is separate from the data control usage. When you select the **Portal** project, data control definition files are created in its sub folder, **Application Sources** and not in the **ViewController** project in which the user interface is created.

2. In the Application Resources panel, expand the **Connections** folder in which you created the repository connection.

3. Drag the repository connection that you intend to use for the new data control and drop it into the Data Controls panel.

   Alternatively, from the **File** menu, choose **New**. In the New Gallery, expand **Business Tier**, select **Content Repository** and then **Content Repository Data Control**, and click **OK**.

   The Create Content Repository Data Control dialog displays with the name of the connection selected in the **Connection Name** list.

   > **Note:** If you created your repository connection under the Resource Palette, then right-click the connection under the Resource Palette and choose **Create Data Control**. From the Resource Palette, you can also drag and drop the required connection onto the Data Controls panel.
   >
   > To add a new connection, click the **Create new content repository connection** icon to display the Create Content Repository Connection dialog. For more information, see Section 27.2, "Configuring Content Repository Connections".

4. In the **Data Control Name** field, enter a name for your data control, for example `MyDataControl`.

5. To add custom attributes, click **Add**. Then, enter a name for the attribute as it should appear in the Data Controls panel, select its type, and enter the JCR path.

   In Content Server, a metadata attribute named `dPropertyName` is mapped in the adapter as shown here:

   ```
   jcr:content/idc:metadata/idc:dPropertyName
   ```

   For example, if the property name is `dWorkflowState`, then it is mapped as `jcr:content/idc:metadata/idc:dWorkflowState`.

   In Oracle Portal, a metadata attribute is mapped in the adapter as shown here:

   ```
   portal:name of the attribute in Oracle Portal
   ```

Note: To retrieve the JCR paths of item attributes, run the `getAttributes` method on the required items. Then reenter the wizard to include those paths by right-clicking the respective data control in the Data Controls panel and choosing **Edit Definition**.

6. Click **OK**.

Note: The data controls include the predefined node types of JCR. These basic node types represent folders (`nt:folder`) and files (`nt:file`). These are derived from `nt:hierarchyNode`, which is a child of the `nt:base` supertype. Additional node types can be derived from the basic types to customize the experience to a particular repository.

28.2.2 What Happens When You Configure a Content Repository Data Control

When the data control is successfully configured, it shows under the Data Controls panel. Expand it in the Data Controls panel to see the hierarchical list of methods, parameters, and operations for the new data control, as shown in Figure 28–1. For detailed information on using the Data Control panel, see Chapter titled "Using Oracle ADF Model in A Fusion Web Application" of *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Figure 28–1 Data Controls Panel - Content Server

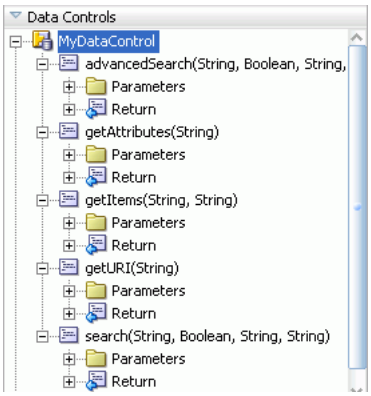


Table 28–8 describes the icons and object hierarchy of a data control.

Table 28–8 The Data Controls Panel Icons and Object Hierarchy

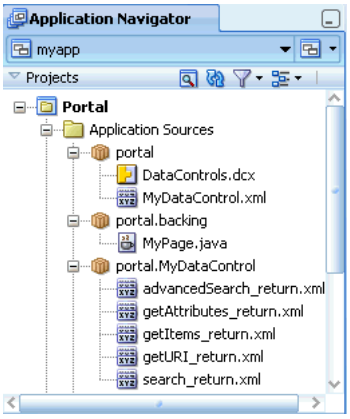
Icon	Name	Description	Used to Create...
	Data Control	Represents a data control. You cannot use the data control itself to create UI components, but you can use any of the child objects listed under it. Depending on how your business services were defined, there may be multiple data controls.	Serves as a container for the other objects, and is not used to create anything
	Method	Represents an operation in the data control or one of its exposed structures that may accept parameters, perform some business logic and optionally return single value, a structure or a collection of those	Command components For methods that accept parameters: command components and parameterized forms
	Method Return	Represents an object that is returned by a custom method. The returned object can be a single value or a collection. A method return appears as a child under the method that returns it. The objects that appear as children under a method return can be attributes of the collection, other methods that perform actions related to the parent collection, and operations that can be performed on the parent collection.	For single values: text fields and selection lists For collections: forms, tables, trees, and range navigation components When a single-value method return is dropped, the method is not invoked automatically by the framework. A user either has to also create an invoke action as an executable, or drop the corresponding method as a button to invoke the method.
	Attribute	Represents a discrete data element in an object (for example, an attribute in a row). Attributes appear as children under the collections or method returns to which they belong. Only the attributes that were included in the view object are shown under a collection. If a view object joins one or more entity objects, that view object's collection will contain selected attributes from all of the underlying entity objects.	Label, text field, date and selection list components
	Operation	Represents a built-in data control operation that performs actions on the parent object. Data control operations are located in an Operations folder under collections or method returns, and also under the root data control node. The operations that are children of a particular collection or method return operate on those objects only, while operations under the data control node operate on all the objects in the data control. If an operation requires one or more parameters, they are listed in a Parameters folder under the operation.	UI components such as buttons or links
	Parameter	Represents a parameter value that is declared by the method or operation under which it appears. Parameters appear in the Parameters folder under a method or operation.	Label, text, and selection list components

The following files (Figure 28–2) are created under the **Portal** project:

- `DataControls.dcx`: Oracle JDeveloper creates this file the first time a data control is created. This file lists all the Oracle ADF data controls created under the current project. This file is required to initialize the data control.
- *`datacontrolname`*`.xml`: This file includes attributes, accessors, and operations of a data control.

- `advancedSearch_return.xml`: This file includes the return type definition for the `advancedSearch` method.
- `getAttributes_return.xml`: This file includes the return type definition for the `getAttributes` method.
- `getItems_return.xml`: This file includes the return type definition for the `getItems` method.
- `getURI_return.xml`: This file includes the return type definition for the `getURI` method.
- `search_return.xml`: This file includes the return type definition for the `search` method.
- `Return.xml`: This file defines the standard operations on collections.

Figure 28–2 Projects Panel - Portal Project



28.3 Editing Content Repository Data Controls

This section describes a generic procedure to edit content data controls that you configured as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".

To edit a content data control:

1. In Oracle JDeveloper, go to the application that contains your content data control.
2. Under Data Controls panel, right-click the data control you intend to edit, and choose **Edit Definition**, as shown in Figure 28–3. The Edit Content Repository Data Control dialog displays, as shown in Figure 28–4.

Figure 28–3 Edit Data Controls

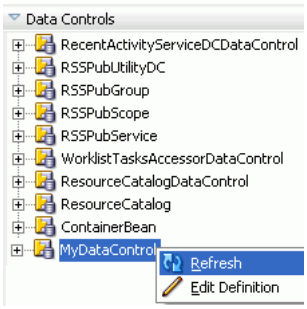
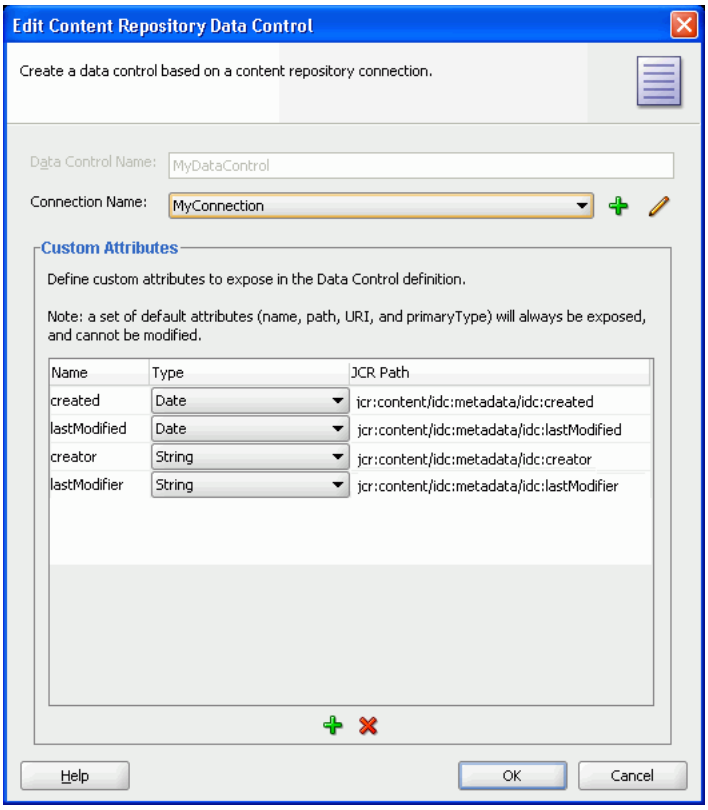


Figure 28–4 Edit Content Repository Data Control Dialog



3. To select a different connection, use the **Connection Name** list.
4. To add a connection, click the **Create new content repository connection** icon to display the Create Content Repository Connection dialog. Depending on the type of connection that you intend to create, see relevant sections in Section 27.2, "Configuring Content Repository Connections" for information about its configuration parameters.
5. To edit an existing connection, click the **Edit selected content repository connection** icon to display the Edit Content Repository Connection dialog. Depending on the type of connection that you intend to modify, see relevant sections in Section 27.2, "Configuring Content Repository Connections" for information about its configuration parameters.
6. To add or remove custom attributes, use the **Add new attribute** icon and the **Remove selected attribute** icon respectively in the Custom Attributes box.
7. Click **OK** to apply the changes.

28.4 Securing a Content Repository Data Control

You can enable security for your content repository connections. For information, see Chapter 69, "Securing Your WebCenter Portal: Framework Application".

28.5 Integrating Content Using Content Data Controls

In this section, you will use `getURI`, `getItems`, `search`, and `advancedSearch` methods of your data control. The basic procedure to use any data control method is to drag and drop it onto a page. When a method is dropped, its ADF Faces tag is added

to the source of the page, and the tag is displayed in the Structure window. For example, dropping the URI attribute of `getURI` as a Go Link adds an `af:goLink` to the page. The destination of the `af:goLink` is set to the EL expression `#{bindings.URI.inputValue}`. This EL expression ties the destination value of the `af:goLink` to the binding container's URI value.

In the page definition file, `methodIterator` is added to the `executables` element, and `methodAction` is added to the `bindings` element. The `methodIterator` and `methodAction` elements define which data control and method is to be used. The `NamedData` attribute of the `methodAction` defines what input parameter values should be provided to this method call. The path and type parameters are common across all data control methods. However, `NamedData` for search and `advancedSearch` includes additional parameters that are unique to those methods. For example, in the `advancedSearch` method, the `NamedData` can also be used to define the search predicates, whether the search is recursive, and so on.

The page definition includes the `AttrNames` element, which defines the attributes of the items available in the bindings container to ADF. These items are based on default attributes and custom attributes that were defined when creating the data control.

For detailed information on using the Data Control panel and working with page definition files, see the chapter "Using Oracle ADF Model in A Fusion Web Application" of *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

The examples in this section use the following data control methods:

- `getURI` to add textual and clickable image links to the repository folder.
- `getItems` to publish the repository content in an ADF table and tree.
- `search` and `advancedSearch` to display those items of the repository that match the search criteria.

This section includes the following subsections:

- Section 28.5.1, "How to Publish Content As Links"
- Section 28.5.2, "What Happens at Runtime"
- Section 28.5.3, "How to Publish Content in a Table"
- Section 28.5.4, "What Happens at Runtime"
- Section 28.5.5, "How to Publish Folder Content in a Tree"
- Section 28.5.6, "What Happens at Runtime"
- Section 28.5.7, "How to Add Search Capabilities to Content Repositories"
- Section 28.5.8, "What Happens at Runtime"
- Section 28.5.9, "What You May Need to Know When Using Search Capabilities"

28.5.1 How to Publish Content As Links

This section describes how to create hyperlinks to files stored in a file system and convert them into textual and image links. You use the **ADF Go Link** and the `getURI` method to create textual links and the Image of **ADF Faces** to create image links.

This section includes the following procedures:

- Section 28.5.1.1, "Publishing Content As a Textual Link"
- Section 28.5.1.2, "Creating a Clickable Image to Link to a Document"

Before you begin:

1. Configure a repository connection as described in Section 27.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".
3. Create a page as described in Section 5.3, "Adding Pages to a Portal."

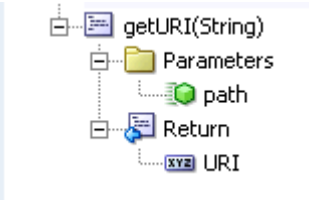
28.5.1.1 Publishing Content As a Textual Link

In this section, you will use the **Oracle ADF Go Link** option of the `getURI` method to publish your content as a textual link. You will also create a link to show a specific item of a repository.

To publish content as a textual link:

1. In the Application Navigator, double-click a page, for example `myPage.jspx`, to open it in the visual editor.
2. In the Data Controls panel, under the repository connection, expand the **getURI (String)** method and expand **Return**. You should see the `URI` attribute, as shown in Figure 28–5.

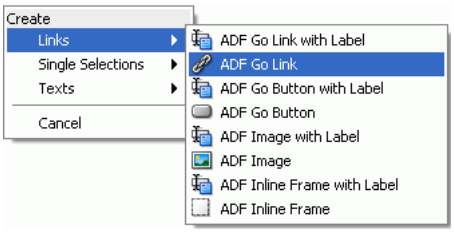
Figure 28–5 The URI Attribute of the getURI Method



3. To create a textual link, select the **URI** attribute and drop it on to the page, or in the Structure window under `af:form`. From the **Create** menu, choose **Links** and then **ADF Go Link**, as shown in Figure 28–6.

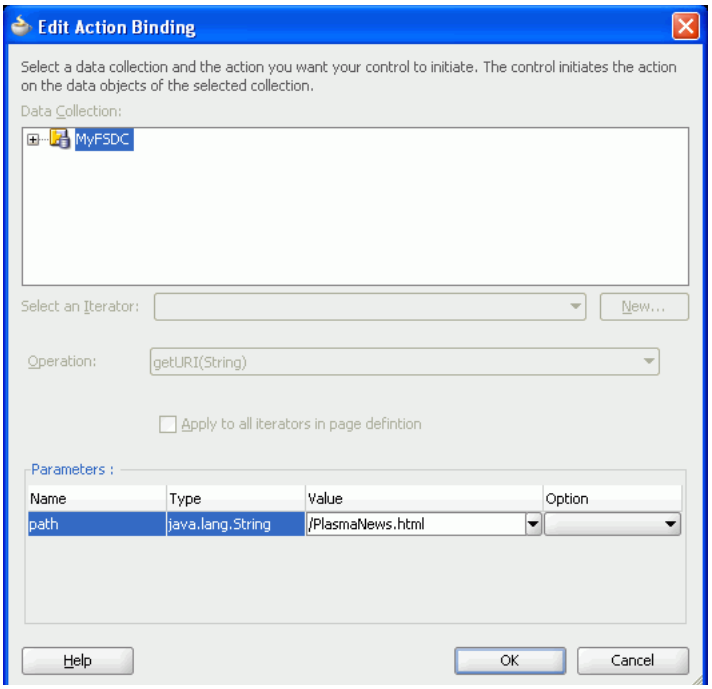
   If this is the first time you have dropped a node onto the page, then the Edit Action Binding dialog displays.

Figure 28–6 Oracle JDeveloper Context Menu for the getURI method



4. In the **Value** field of the **path** parameter, enter the path of the file for which you intend to create the link, as shown in Figure 28–7. You must enter a leading slash (`/`), for example `/PlasmaNews.html`. To modify or delete this path later, click the arrow icon next to the node in design mode and choose **Go to Binding** from the context menu.

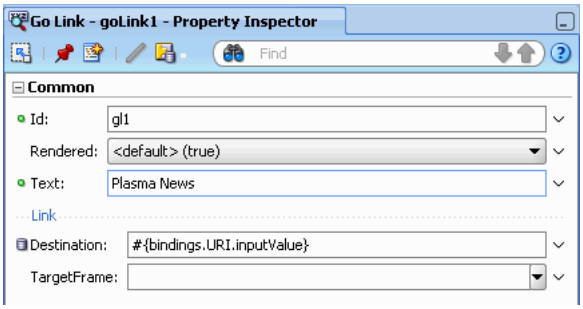
Figure 28–7 Edit Action Binding



> **Note:** To grant edit, personalize, customize, and view permissions at the attribute level, see Section 28.4, "Securing a Content Repository Data Control".

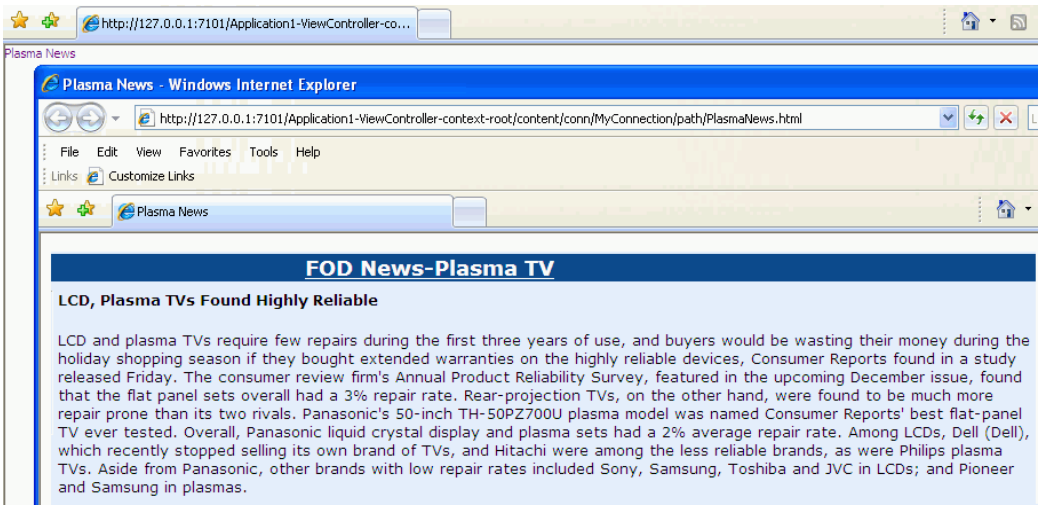
5. Click **OK**.
6. Right-click the page and select **Run**. In your browser, you should see the URL for the file path entered in the Edit Action Binding dialog without any formatting.
7. By default, the link displays the text `goLink1`. In the Structure window, select **af:goLink - goLink1** and view the properties in the Property Inspector.
8. In the **Text** field, enter a name for the link, for example, `Plasma News`, as shown in Figure 28–8.

Figure 28–8 Go Link Properties



9. Right-click your page and choose **Run**. The page appears in your browser window with the new link, as shown in Figure 28–9.

Figure 28–9 ADF Go Link in a Browser



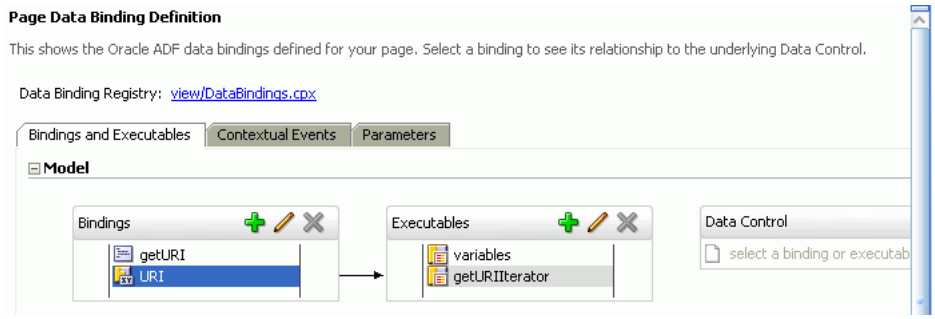
10. Click the link to check that the correct file is displayed.

In Section 28.5.1.2, "Creating a Clickable Image to Link to a Document", you will extend this textual link into an image link.

To add a link to another item, in the same JSF page:

1. In the Application Navigator, right-click the page in which you created the ADF Go Link, and choose **Go to Page Definition**. The Page Data Binding Definition displays in the design view, as shown in Figure 28–10.

Figure 28–10 Page Data Binding Definition



2. Go to the source view of the page definition.

3. In the `executables` element, add another `methodIterator` and change the `methodIterator id` and value of `Binds`, as shown in **Bold** in the following example:

```
<executables>
  <variableIterator id="variables"/>
  <methodIterator Binds="getURI.result" DataControl="MyDataControl"
     RangeSize="25" BeanClass="portal.MyDataControl.getURI_return"
     id="getURIIterator"/>
  <methodIterator Binds="getURI1.result" DataControl="MyDataControl"
    RangeSize="25" BeanClass="portal.MyDataControl.getURI_return"
    id="getURIIterator1"/>
  </executables>
```

4. In the `Bindings` element, add another `methodAction` and change the `methodAction id`, `ReturnName`, and `NDValue`, as shown in **Bold** in the following example:

```
<methodAction id="getURI"
  RequiresUpdateModel=true Action="invokeMethod" MethodName="getURI"
  IsViewObjectMethod="false" DataControl="MyDataControl"
  InstanceName="MyDataControl"
  ReturnName="MyDataControl.methodResults.getURI_MyDataControl_getURI_result">
  <NamedData NDName="path" NDValue="/PlasmaNews.html"
    NDType="java.lang.String"/>
</methodAction>
<methodAction id="getURI1"
  RequiresUpdateModel="true" Action="invokeMethod" MethodName="getURI"
  IsViewObjectMethod="false" DataControl="MyDataControl"
  InstanceName="MyDataControl"
  ReturnName="MyDataControl.methodResults.getURI_MyDataControl_getURI1_result">
  <NamedData NDName="path" NDValue="/FusionOrderDemoLogo.jpg"
    NDType="java.lang.String"/>
</methodAction>
```

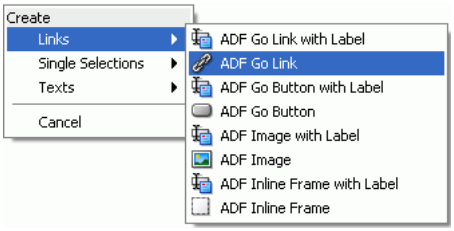
5. In the `Bindings` element, add another `attributeValues` tag to specify the new Id, as shown in **bold** in the following example:

```
<attributeValues
      IterBinding="getURIIterator"
      id="URI">
   <AttrNames>
      <Item Value="URI"/>
   </AttrNames>
</attributeValues>
<attributeValues
        IterBinding="getURIIterator1"
        id="URI1">
  <AttrNames>
        <Item Value="URI"/>
  </AttrNames>
</attributeValues>
```

6. In the Data Controls panel, under the repository connection, expand the **getURI (String)** method and expand **Return**. You should see the `URI` attribute.

7. To create a textual link, select the **URI** attribute and drop it on to the page, or in the Structure window under `af:form`. From the **Create** menu, choose **Links** and then **ADF Go Link**, as shown in Figure 28–11.

Figure 28–11 Oracle JDeveloper Context Menu for the getURI method

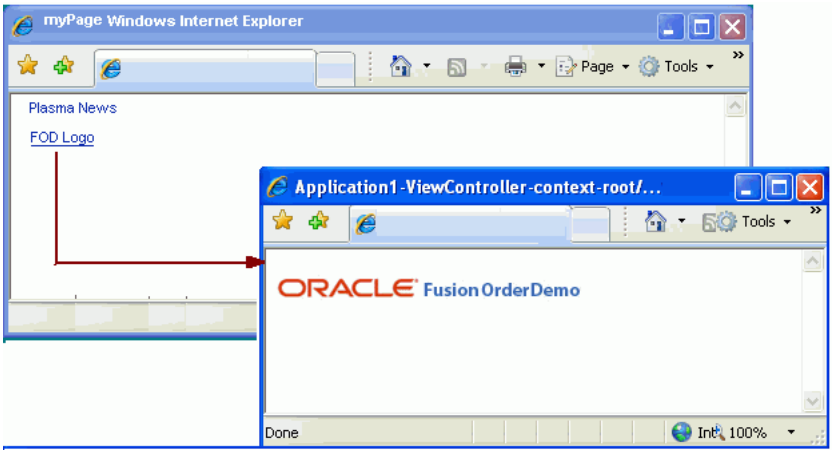


8. In the Structure window, double-click the new goLink; for example, **af:goLink2** to display the Go Link Properties window.

9. In the **Text** field, enter a display name for your new link; for example, `FOD Logo`.

10. In the **Destination** field, click the down arrow to display the Expression Builder dialog. Then, expand **ADF Bindings**, **bindings**, and **URI1**.
11. Double-click **inputValue** variable to create the `#{bindings.URI1.inputValue}` expression, and click **OK**. This expression is based on new elements that you added in `executables` and `bindings`.
12. Run your page. The new link should display content from the new path (`NDValue`) that you specified in step 4. Figure 28–12 shows the new link, `FOD Logo`, in addition to the `Plasma News` link that was added in the first part of Section 28.5.1.1, "Publishing Content As a Textual Link".

Figure 28–12 New Textual Link



You can add as many links as required by following these steps.

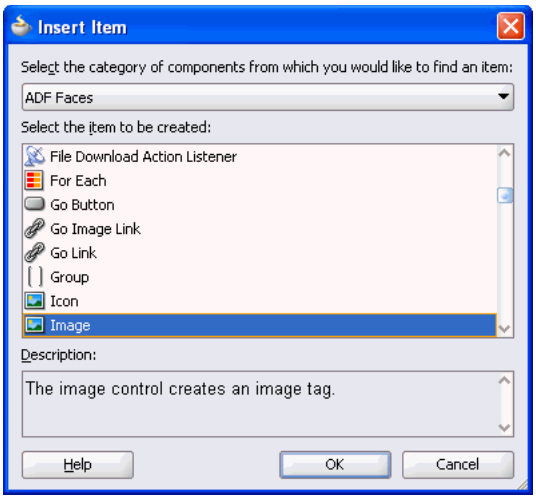
28.5.1.2 Creating a Clickable Image to Link to a Document

In this section, you will use the Image option of ADF Faces to publish a document as a clickable image, that is, clicking the image object will display your document.

To publish content as a clickable image object:

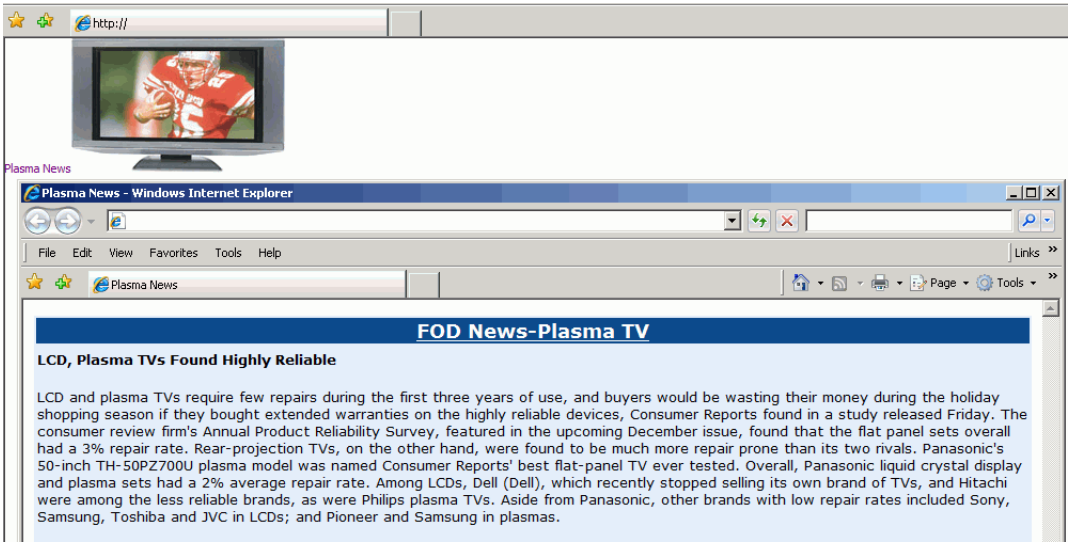
1. In the Application Navigator, open the page, in which you created the ADF Go Link, by double-clicking it.
2. To convert the textual link that you created in the first part of Section 28.5.1.1, "Publishing Content As a Textual Link", in the Structure window, right-click **af:goLink - goLink1** and choose **Browse**. The Insert Item dialog displays.
3. From the **Select the item to be created** list, select **Image**, as shown in Figure 28–13.

Figure 28–13 Insert ADF Faces Item



4. Click **OK**. The Insert Image dialog displays.
5. Browse to the image file you intend to display as the image link; for example, `plasma.jpg`.
6. Click **Finish**.
7. Right-click your page and choose **Run**. Figure 28–14 shows a sample output.

Figure 28–14 ADF Object Image Link in a Browser



28.5.2 What Happens at Runtime

In the preceding examples, you created hyperlinks to files that are stored in a file system using **ADF Go Link** and converted them into textual and image links using the `getURI` method and the **Image** component of **ADF Faces**.

At runtime, the ADF framework uses the information from the page definition file to invoke the data control methods required by the page. For information about the page lifecycle, see the chapter "Understanding the Fusion Page Lifecycle" of *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

At runtime, the `getURI` data control method is invoked to convert the given path into a valid HTTP URI for the content repository for which the data control is configured. The returned URL is syntactically correct for the target repository, but is not guaranteed to find a resource, that is, the `getURI` method does not validate that the path corresponds to an existing JCR node. When the page is rendered and the clickable image or link is clicked, the application's get handler converts the HTTP URL into a JCR path and attempts to retrieve the content for the JCR path. The get handler also supplies `mimeType` information in the response `Content-Type` header, if such information is available from the repository. This is because in JCR, `jcr:mimeType` is an optional property of the `nt:resource` node type.

28.5.3 How to Publish Content in a Table

In this section, the `getItems` data control method is used to publish file and folder information in a table. This section describes the following procedures:

- Section 28.5.3.1, "Displaying Files and Folders in Read-Only Format"
- Section 28.5.3.2, "Displaying the Name Attribute As a Go Link"
- Section 28.5.3.3, "Configuring a Table to Show Only Files"

Before you begin:

1. Configure a repository connection as described in Section 27.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".
3. Create a page as described in Section 5.3, "Adding Pages to a Portal."

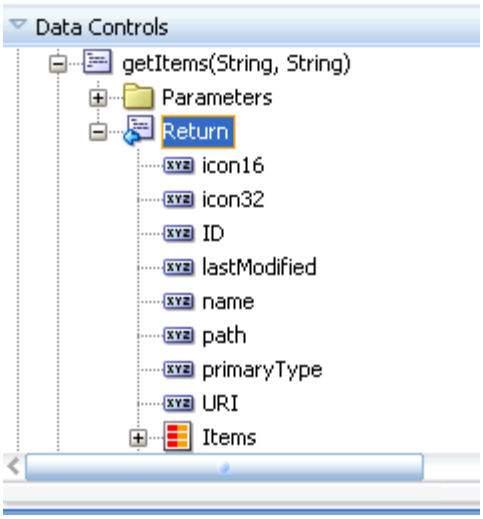
28.5.3.1 Displaying Files and Folders in Read-Only Format

Here you will create a read-only ADF table using the `getItems` method.

To display folder content in a read-only table:

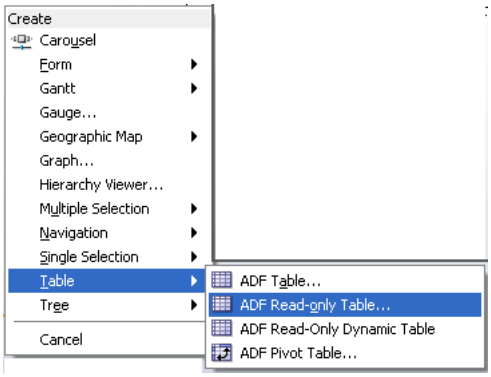
1. To open the page in the visual editor, double-click it in the Application Navigator.
2. In the Data Controls panel, under your data control, expand the **getItems** method and the **Return** node, as shown in Figure 28–15.

Figure 28–15 The Return node of the getItems method



3. To create a table that lists every file and folder available through this data control, drop the **Return** node on to the page or under `af:form` in the Structure window. From the **Create** menu, choose **Table** and then **ADF Read-only Table** , as shown in Figure 28–16.

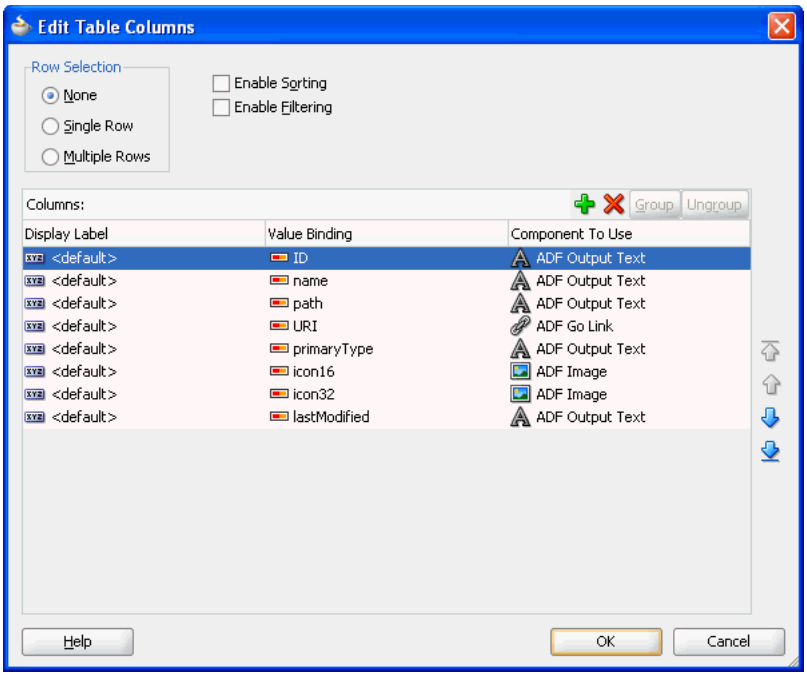
Figure 28–16 The Create Context Menu for getItems Method



4. In the Edit Table Columns dialog, as shown in Figure 28–17, Select the row for the `getItems.name` **Value Binding** and enter an appropriate value for the **Display Label**; for example, `Name`.

   Repeat this step for the **path**, **URI**, **primaryType**, and **lastModified** attributes. Enter new display labels such as `name`, `Location`, `URL`, and so on, then click **OK.**

Figure 28–17 Edit Table Columns

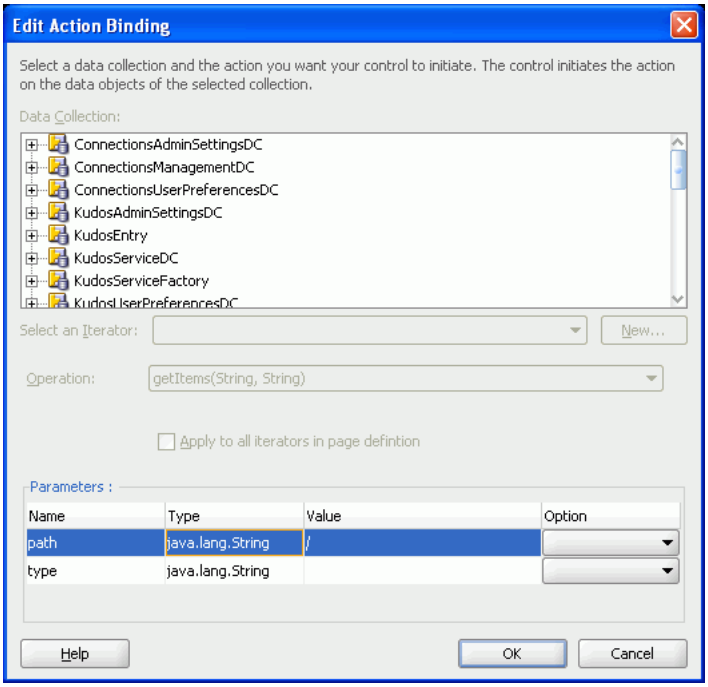


5. If this is the first time you have dropped a node onto the page, the Edit Action Binding dialog displays.

   In the Edit Action Binding dialog, enter the path of the content directory as the `path` parameter, as shown in Figure 28–18. You must enter a leading slash (`/`). To modify or delete this path later, click the arrow icon next to the node in design mode and choose **Go to Binding** from the context menu.

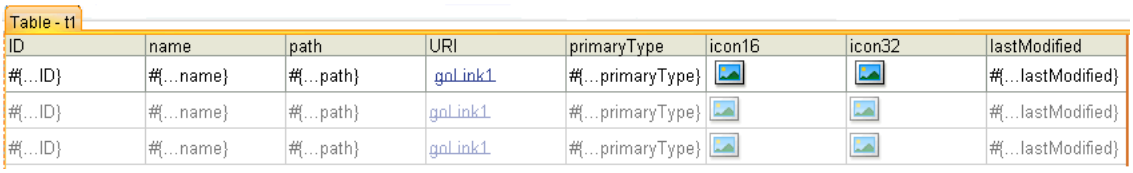
   Leave the `type` parameter blank. This implies that the table displays both files and folders.

Figure 28–18 Edit Action Binding



6. Click **OK.** You should now see a table on the page that looks like Figure 28–19.

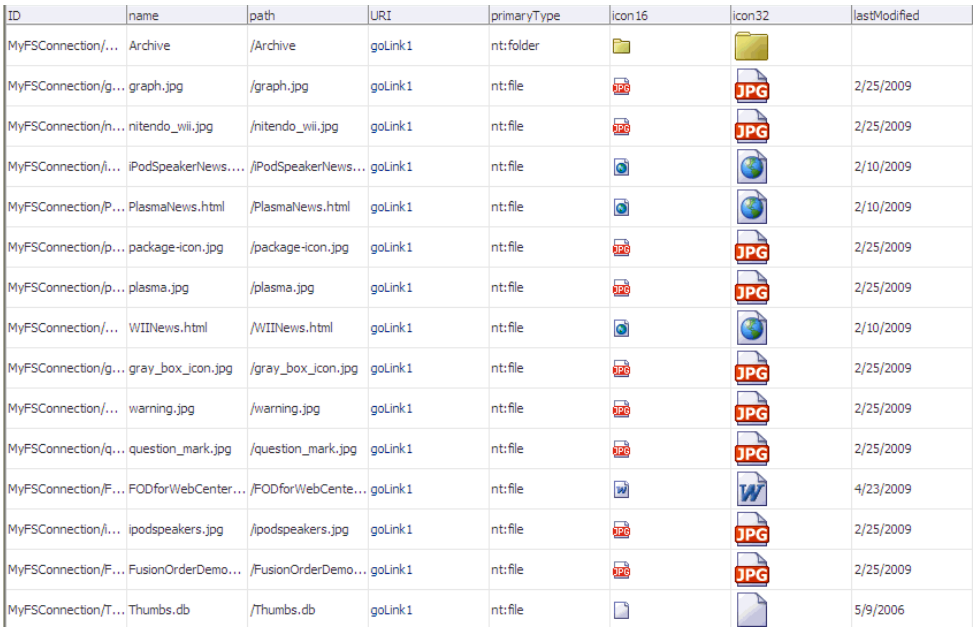
Figure 28–19 Read-Only Table for Publishing Folder Content



> **Note:** You can turn the page caching on or off. To do so, open the page definition and expand **executables**, and select **getItemsIterator** in the Structure window. Then, in the Property Inspector, set **CacheResults** to **true** or **false**, as required.

7. Run the page. You should see a list of all files and folders available in your content directory. Figure 28–20 shows a read-only table displaying both files and folders at runtime.

Figure 28–20 Files and Folders Displayed in a Read-Only Table



ID	name	path	URI	primaryType	icon16	icon32	lastModified
MyFSConnection/...	Archive	/Archive	goLink1	nt:folder			
MyFSConnection/g...	graph.jpg	/graph.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/n...	nitendo_wii.jpg	/nitendo_wii.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/i...	iPodSpeakerNews....	/iPodSpeakerNews...	goLink1	nt:file			2/10/2009
MyFSConnection/P...	PlasmaNews.html	/PlasmaNews.html	goLink1	nt:file			2/10/2009
MyFSConnection/p...	package-icon.jpg	/package-icon.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/p...	plasma.jpg	/plasma.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/...	WIINews.html	/WIINews.html	goLink1	nt:file			2/10/2009
MyFSConnection/g...	gray_box_icon.jpg	/gray_box_icon.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/...	warning.jpg	/warning.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/q...	question_mark.jpg	/question_mark.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/F...	FODforWebCenter...	/FODforWebCente...	goLink1	nt:file			4/23/2009
MyFSConnection/i...	ipodspeakers.jpg	/ipodspeakers.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/F...	FusionOrderDemo...	/FusionOrderDemo...	goLink1	nt:file			2/25/2009
MyFSConnection/T...	Thumbs.db	/Thumbs.db	goLink1	nt:file			5/9/2006

By default, the table displays file or folder attributes as read-only text (`af:outputText`). The next section describes how to display the `Name` attribute (`name`) as a Go Link (`af:goLink`).

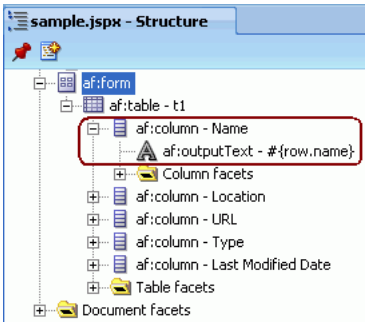
28.5.3.2 Displaying the Name Attribute As a Go Link

In this section, you will convert the `Name` attribute of the table that you created in Section 28.5.3.1, "Displaying Files and Folders in Read-Only Format" into a link using an **ADF Go Link** component. You will also configure the table to show only the **Name** column.

To display the Name attribute as a Go Link:

1. In the Structure window (Figure 28–21), expand the first column of the table (`af:column - Name`) to show the default display format `af:outputText - #{row.name}`.

Figure 28–21 Default Formatting for the Name Column



2. In the source view for the page, change the name column to use a go link. The link displays the name but links to the URI of the entry:

   Replace the `af:outputText` in **bold**

```
<af:column sortProperty="name" sortable="false"
```

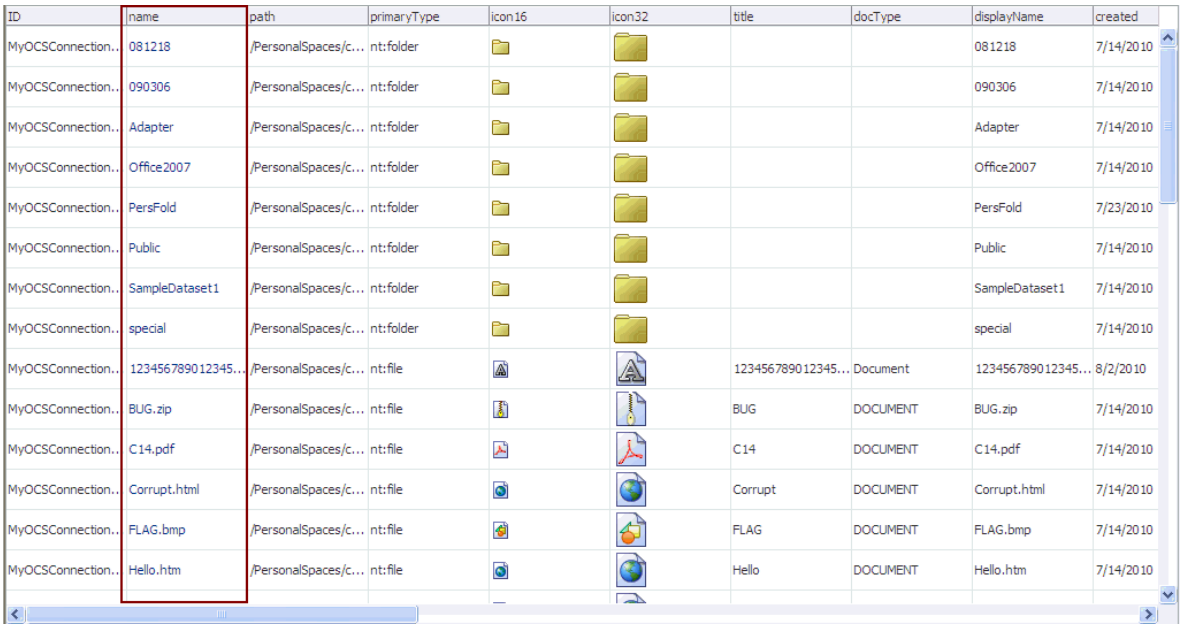
```
      headerText="#{bindings.Return.hints.name.label}" id="c2">
   <af:outputText value="#{row.name}" id="ot1"/>
</af:column>
```

with an `af:goLink` as shown in **bold**:

```
af:column sortProperty="name" sortable="false"
     headerText="#{bindings.Return.hints.name.label}"
     id="c2">
  <af:goLink text="#{row.name}" id="goLink2"
     destination="#{row.bindings.URI.inputValue}"/>
</af:column>
```

3. In the Structure window, right-click `af:column - URL` and select **Delete**.
4. Right-click the page in the Application Navigator and choose **Run**. You should see a list of hyperlinked file and folder names like the one shown in Figure 28–22. This figure shows the `name` attribute as a link. Clicking a link in the **URI** column opens the respective file or folder and shows its contents.

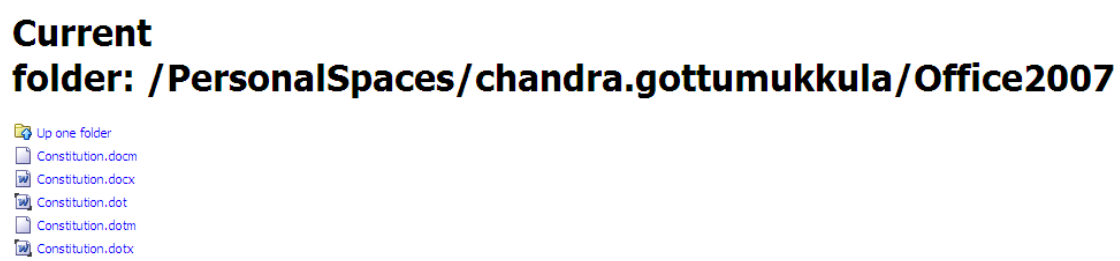
Figure 28–22 Folder Content Displayed as Hyperlinks



ID	name	path	primaryType	icon16	icon32	title	docType	displayName	created
MyOCSConnection..	081218	/PersonalSpaces/c...	nt:folder					081218	7/14/2010
MyOCSConnection..	090306	/PersonalSpaces/c...	nt:folder					090306	7/14/2010
MyOCSConnection..	Adapter	/PersonalSpaces/c...	nt:folder					Adapter	7/14/2010
MyOCSConnection..	Office2007	/PersonalSpaces/c...	nt:folder					Office2007	7/14/2010
MyOCSConnection..	PersFold	/PersonalSpaces/c...	nt:folder					PersFold	7/23/2010
MyOCSConnection..	Public	/PersonalSpaces/c...	nt:folder					Public	7/14/2010
MyOCSConnection..	SampleDataset1	/PersonalSpaces/c...	nt:folder					SampleDataset1	7/14/2010
MyOCSConnection..	special	/PersonalSpaces/c...	nt:folder					special	7/14/2010
MyOCSConnection..	123456789012345...	/PersonalSpaces/c...	nt:file			123456789012345...	Document	123456789012345...	8/2/2010
MyOCSConnection..	BUG.zip	/PersonalSpaces/c...	nt:file			BUG	DOCUMENT	BUG.zip	7/14/2010
MyOCSConnection..	C14.pdf	/PersonalSpaces/c...	nt:file			C14	DOCUMENT	C14.pdf	7/14/2010
MyOCSConnection..	Corrupt.html	/PersonalSpaces/c...	nt:file			Corrupt	DOCUMENT	Corrupt.html	7/14/2010
MyOCSConnection..	FLAG.bmp	/PersonalSpaces/c...	nt:file			FLAG	DOCUMENT	FLAG.bmp	7/14/2010
MyOCSConnection..	Hello.htm	/PersonalSpaces/c...	nt:file			Hello	DOCUMENT	Hello.htm	7/14/2010

5. Click a file name. The file you pick should appear in a browser window.
6. Click the name of a folder, the contents of the folder display, as shown in Figure 28–23.

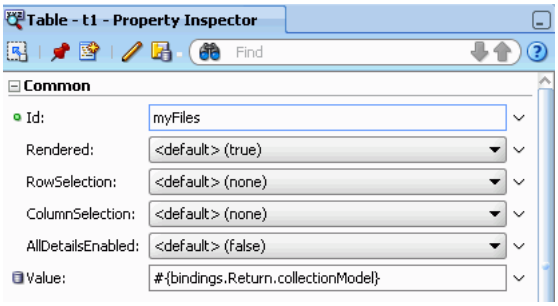
Figure 28–23 Folder Contents



To configure the table to show only the Name column:

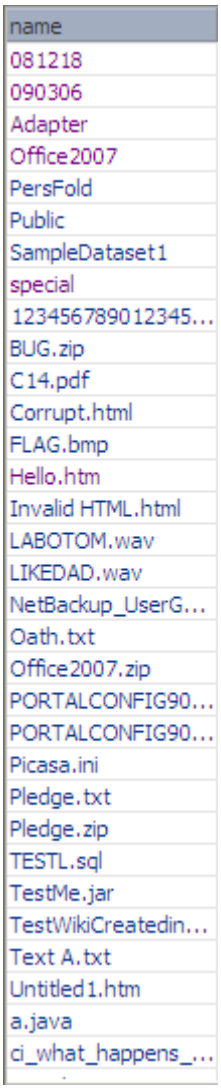
1. In the Structure window, under the **af:table - t1** node, delete all but the **Name** column.
2. Double-click the **af:table - t1** node to display the Property Inspector.
3. Under the **Common** tab, in the **Id** field, enter a name, for example `myFiles`, as shown in Figure 28–24.

Figure 28–24 Table Properties - Common Tab



4. Run the page to view the output. Figure 28–25 shows only one column of the table since other rows were removed from the Structure window at design time. This table shows both files and folders, because you left the `type` parameter blank when creating the table.

Figure 28–25 Files and Folders Displayed in a Single-Column Table



In the next section, the **Name** column will be configured to show only files.

28.5.3.3 Configuring a Table to Show Only Files

The `type` attribute is used to configure a table to show only files and not folders.

To configure the table to show files:

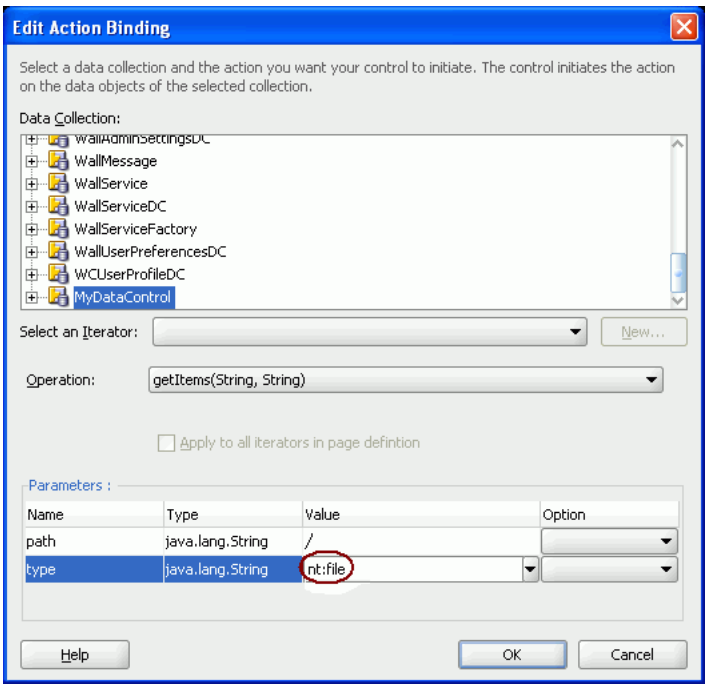
1. Right-click your page and choose **Go to Page Definition**.

   > **Note:** The `RangeSize` binding setting, which is used to control the number of items displayed on a page, is set to `10` by default in the page definition file. You can change it, as required, in the Property Inspector.

2. In the **Overview** tab, double-click **getItems** under **Bindings**. The Edit Action Binding dialog displays.

3. The **type** options are `nt:file` and `nt:folder`. To specify the display of only files, enter `nt:file` under the **Value** column, as shown in Figure 28–26, and click **OK**.

Figure 28–26 Display Files Only



4. Now run the page. Figure 28–27 shows only files in the single-column table because the column type is `nt:file`.

Figure 28–27 Files Displayed in a Single-Column Table



28.5.4 What Happens at Runtime

The `getItems` method of the JCR data control retrieves the child items of a JCR folder (type `nt:folder`). This method is called with a path to a folder and optionally a type to which the returned child nodes are restricted.

The `path` parameter must be the path of a folder. An exception to this rule is that the root of the repository does not have to be a folder. Hence, the `getItems` method can retrieve the child items for a path that corresponds to a folder, or that is the root of the repository. Otherwise the `getItems` method does not attempt to retrieve the child items and therefore the result set is never populated.

At runtime, the JCR data control calls the `Session.getItem` method. This method returns a JCR node. The data control then calls the `node.getNodes` to retrieve child nodes. The child nodes are filtered according to the specified type; for example, the `nt:file` type. The data control passes on these child nodes to ADF. The data control ensures that the JCR node object is adapted to ADF so that JCR properties are available as data control item attributes that can be consumed through the bindings container.

In the example, each row in the collection returned by the data control is stored as a row in the `af:table` table. However, a column is only displayed for each `af:column` defined in the `af:table`, and not for every possible ADF item attributes returned by the data control. In the first part of the example, the drag and drop action creates an `af:column` for every ADF attribute available for each ADF item returned by the data control.

If the `af:table` is modified to include only the `af:column` for the name, then at runtime it only requests the name of the ADF item. That is, the data control runs a method to fetch the name of the JCR node. At design time, when the name column is converted to an `af:goLink`, the destination of **Go Link** destination is set to the **URI** value of the item, as shown in the following syntax:

```
<af:goLink text="#{row.name}" destination="#{row.URI}"/>
```

At runtime, for each ADF item in the table, the data control runs methods to return both the JCR name of the item and its HTTP URL.

28.5.5 How to Publish Folder Content in a Tree

In this section, you will use the `getItems` method to publish content in a hierarchal tree format. This section describes the following procedures:

- Section 28.5.5.1, "Displaying Files and Folders in Read-Only Format"
- Section 28.5.5.2, "Displaying File Names As Hyperlinks"

Before you begin:

1. Configure a repository connection as described in Section 27.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".
3. Create a page as described in Section 5.3, "Adding Pages to a Portal."

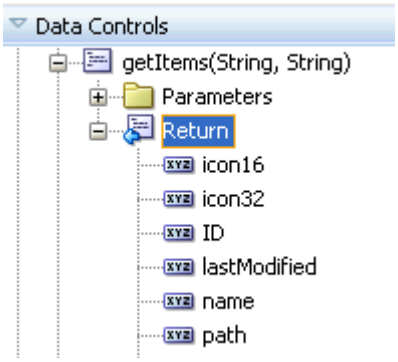
28.5.5.1 Displaying Files and Folders in Read-Only Format

In this section, you will display your content in the tree format.

To display your content in the tree format:

1. In the Application Navigator, double-click your page to open it in the design view.
2. In the Data Controls panel, under your data control, expand the **getItems** method as shown in Figure 28–28.

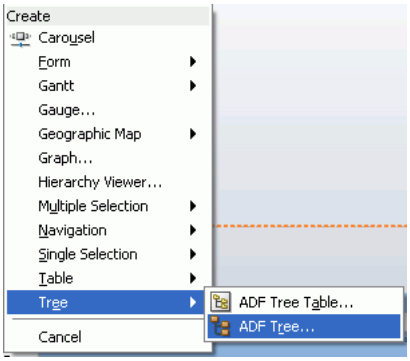
Figure 28–28 Parameters of the getItems Method



3. To display your content as an **ADF Tree**, select the **Return** node and drag it onto the page. From the Create menu, choose **Tree** and then **ADF Tree**, as shown in Figure 28–29.

   If this is the first time you have dropped a node onto the page, the Edit Action Binding dialog displays.

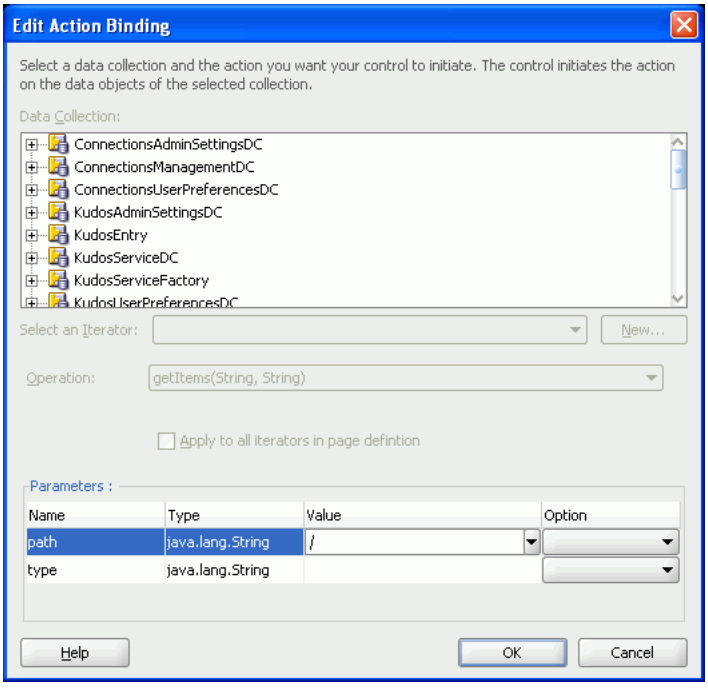
Figure 28–29 Oracle JDeveloper Create Menu for getItems



4. To create a tree that displays everything under the base path, enter the slash (`/`) for the `path` parameter, as shown in Figure 28–30. To modify or delete this path later, click the arrow icon next to the node in the design mode and choose **Go to Binding** from the context menu.

   Leave the `type` parameter blank to show both files and folders.

Figure 28–30 The Edit Action Binding Dialog

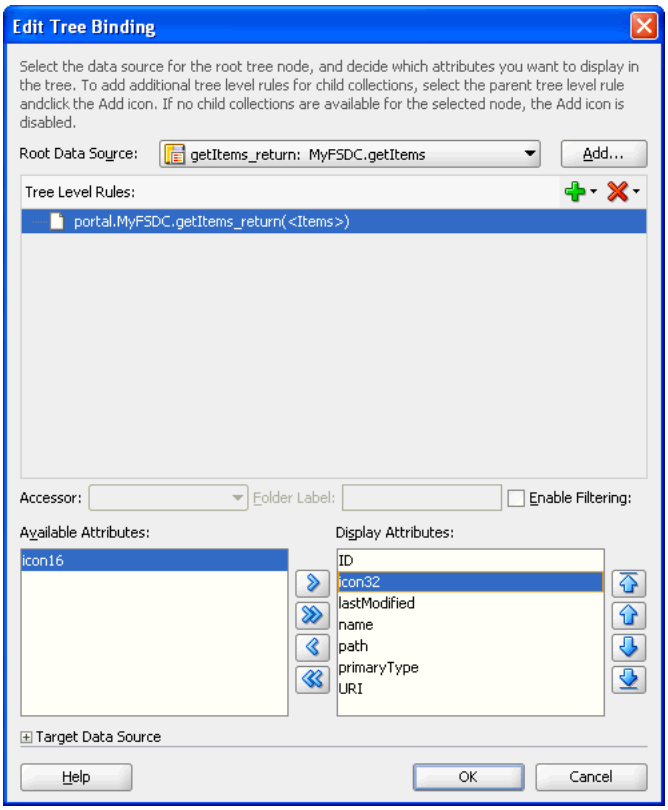


5. Click **OK**. The Edit Tree Binding dialog displays.

6. To show item names, paths, and types at runtime, under **Available Attributes**, select the desired attributes and move them to the **Display Attributes** list.

7. In the **Tree Level Rules** box, click the **Add Rule** icon and select **Items** to create a rule, as shown in Figure 28–31, which enables the tree to find its child items.

Figure 28–31 Edit Tree Binding



8. Click **OK.** A tree displays in the page that looks like Figure 28–32.

Figure 28–32 Tree for Navigating Folder Content



9. Run your page to display the results.

   When the page appears in your browser window, you should see a list of files and folders available through your data control. Figure 28–33 displays a tree of files and folders in the read-only format based on **ADF tree** dropped on the JSF page. Expand a branch to see the content in this subdirectory.

> **Note:** By default, the range size is `10`. To change the number of items displayed in the tree, edit the **RangeSize** property for the data control in the page definition file (*name*`PageDef.xml`).

Figure 28–33 Folder Content Displayed in a Tree



By default, the tree displays file and folder names as read-only text. The next section describes how to create hyperlinks to file names. In the following section, folder names will remain read-only text because they are required for navigation through the tree.

28.5.5.2 Displaying File Names As Hyperlinks

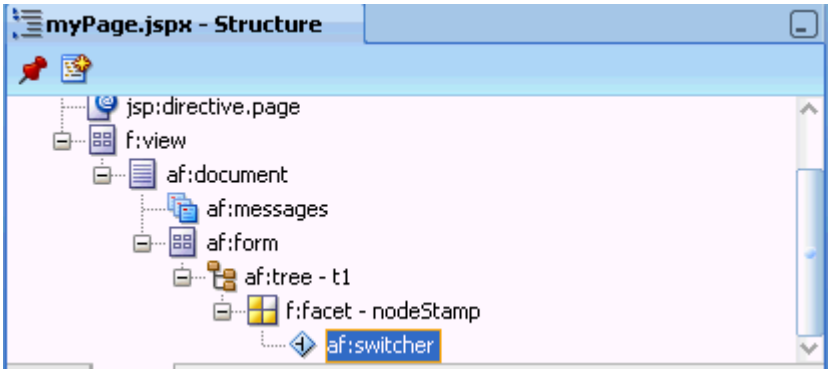
To create hyperlinks to file names and to keep folder names read-only, you need the `af:switcher` component with two facets: one for folders and one for files.

To use the Switcher component for folders and files:

1. In the Structure window, navigate to `f:facet - nodeStamp` and delete `af:outputText-#{node}`.
2. Right-click **nodeStamp** and choose **Insert inside f:facet - nodeStamp** and then **Browse**.
3. In the Insert Item dialog, from the dropdown list for selecting the category, select **ADF Faces**.
4. From the **Select the item to be created** list, select **Switcher**, and click **OK**.

   A Switcher is added under **f:facet - nodeStamp** in the Structure window, as shown in Figure 28–34.

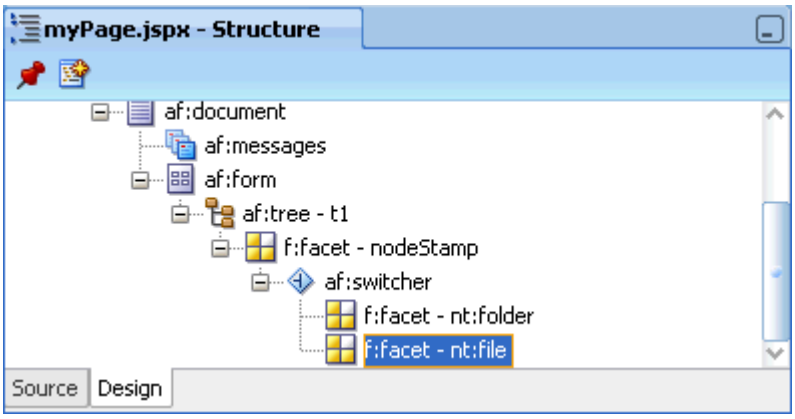
Figure 28–34 Output Text Converted to a Switcher Component



5. Double-click **af:switcher** to display the Property Inspector, if it is not displayed.

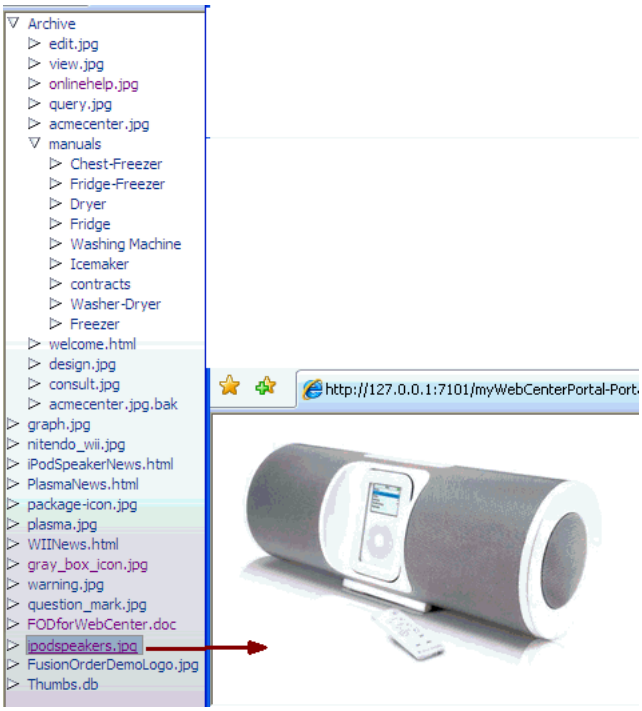
6. Under the **Common** tab, in the **FacetName** field, enter the expression `#{node.primaryType}`.
7. In the Structure window, insert two facets for the switcher. Right-click **af:switcher**, choose **Insert Inside af:switcher** and then **Facet**. The Insert Facet dialog displays.
8. Name the first facet `nt:folder` and click **OK**. Folder names require no additional formatting, so you can display the node names as plain text. Name the second facet `nt:file`. The facets look like Figure 28–35.

Figure 28–35 Switcher Component with Two Facets



9. Right-click **f:facet - nt:folder**, choose **Insert Inside f:facet - nt:folder** and then **Browse**.
10. In the Insert Item dialog, choose **ADF Faces** from the dropdown list.
11. From the **Select the item to be created** list select **Output Text** and click **OK**.
12. Double-click **af:outputText - outputText1** to display the Property Inspector, if it is not displayed already. Under the **Common** tab, in the **Value** field, enter the expression `#{node.name}`.
13. Right-click **f:facet - nt:file**, choose **Insert Inside f:facet - nt:folder** and then **Browse**.
14. In the Insert Item dialog, choose **ADF Faces** from the dropdown list.
15. From the **Select the item to be created** list select **Go Link** and click **OK**.
16. In the Structure window, double-click **af:goLink - goLink** to display the Property Inspector, if it is not already displayed.
17. Under the **Common** tab, in the **Text** field, enter the `#{node.name}` expression.
18. In the **Destination** field, enter the expression `#{node.URI}`.
19. From the **TargetFrame** list, select **_blank**.
20. Run the page. Figure 28–36 displays files names as hyperlinks, because the `nt:file` facet of the switcher under `af:tree` was converted to a link (`node.URI`). Clicking a link displays the respective item.

Figure 28–36 Tree with File Names as Hyperlinks



28.5.6 What Happens at Runtime

The JCR data control `getItems` method is designed to retrieve the child items of a JCR folder (type `nt:folder`). The method is invoked with a path to a folder and optionally a type to which the returned child nodes are restricted. The `path` parameter must be the path of a folder for the `getItems` method to retrieve the child items. An exception to this rule is that the root of the repository does not have to be a folder. Hence, the `getItems` method can retrieve the child items for a path that corresponds to a folder, or that is the root of the repository. Otherwise the `getItems` method does not attempt to retrieve the child items and therefore the result set is never populated. If the path is valid, then the data control invokes `JCR Session.getItem()` on this path which returns a JCR node, and then it invokes `node.getNodes()` to retrieve all child nodes. The child nodes are filtered according to the type supplied; for example, to return only the `nt:file` type child node. This is the result that is provided by the data control to ADF. The data control ensures that the JCR node object is adapted to ADF such that JCR properties are available as data control item attributes that can be consumed by way of the bindings container.

The `af:tree` renders each node in the collection returned by the data control as a node in its tree. In the first part of the example, it displays the `name`, `type`, and `URI` for each node. Each of these values is retrieved through the data control. When the switcher is added to the `af:tree` the node's `primaryType` value is used to differentiate how a node is rendered. If the node is of primary type `nt:folder`, then only its name is shown in the tree node. However if the node is of type `nt:file`, then the node renders `go:Link`, the destination of which is the node's URI. The data control `getItems` method is invoked again to retrieve the child nodes of any folder node in the tree.

28.5.7 How to Add Search Capabilities to Content Repositories

With the help of two examples, this section describes how to add simple and advanced search capabilities for the integrated content. The simple search enables users to search for the content based on name or content fragments in specific locations. The advanced search enables users to search by attribute values of the content.

This section contains the following:

- Section 28.5.7.1, "Adding Simple Search Capabilities"
- Section 28.5.7.2, "Adding Advanced Search Capabilities"

Before you begin:

1. Configure a repository connection as described in Section 27.2, "Configuring Content Repository Connections".
2. Configure a data control as described in Section 28.2, "Configuring Content Data Controls for JCR Adapters".
3. Create a page as described in Section 5.3, "Adding Pages to a Portal."

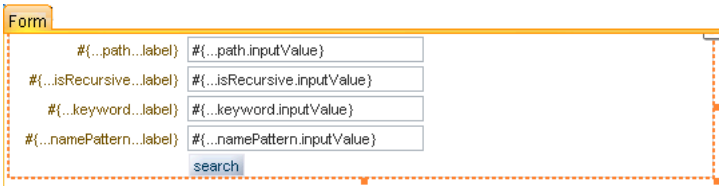
28.5.7.1 Adding Simple Search Capabilities

In this section, you will enable simple search capabilities in your page. This will let you perform wildcard (%) search.

To enable the search function:

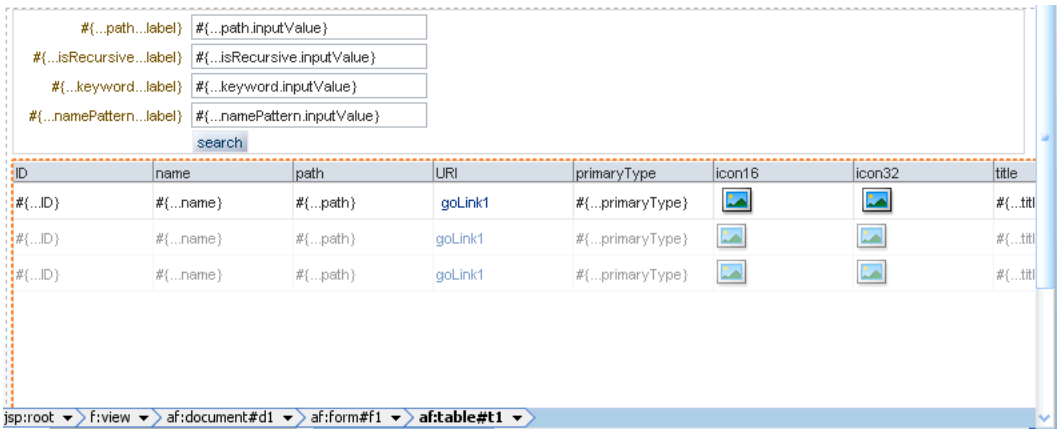
1. In the Application Navigator, double-click your page to open it.
2. In the Data Controls panel, select the **search** node.
3. To enable users to perform a search by clicking a button, drag and drop the **search** node on your page. From the Create menu, choose **Parameters** and then **ADF Parameter Form**. The Edit Form Fields dialog displays.
4. Click **OK**. The ADF parameter form is added to the page, as shown in Figure 28–37

Figure 28–37 ADF Parameter Form in the Design View



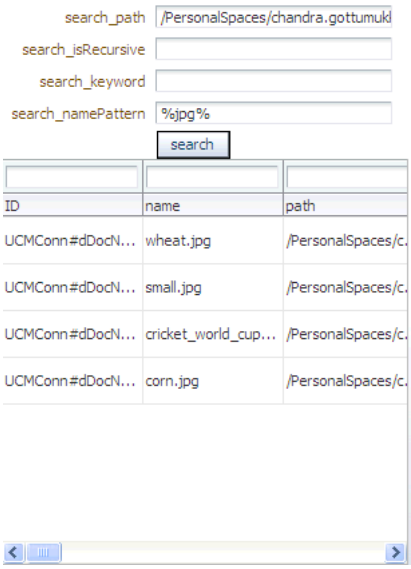
5. To enable the display of search results in a read-only table, drag and drop the **Return** node onto the page. From the **Create** menu, choose **Table** and then **ADF Read-Only Table**. The Edit Table Columns dialog displays.
6. Click **OK**. A table similar to Figure 28–38 displays.

Figure 28–38 Table with Four Columns - search



7. Run this page and specify `/` for **search_path** and `%jpg%` for **namePattern**. All `.jpg` files stored in the root of your repository display, as shown in Figure 28–39.

Figure 28–39 Search Results for .jpg Files



28.5.7.2 Adding Advanced Search Capabilities

In this section, you will add advanced search capabilities to your page that will enable you to perform search based on the last modified dates of items located in your file system repository.

To enable the advanced search function:

1. In the Application Navigator, open your page in which you intend to create the advanced search function. The page must be automatically exposed in new managed bean, where name=`advancedSearch`, class=`AdvancedSearch`, package=`view`. (In the Create JSF Page dialog > Page Implementation > Automatically Expose UI Components in a New Managed Bean)

   In this example the page is called `advancedSearch.jspx`.

2. Double-click the page to open it.

3. In the Component Palette, select **ADF Faces**.
4. From the list of **ADF Faces - Layout** components, drag **Panel Form Layout** onto the page.
5. From the Component Palette, drag **Input Date** into the **Panel Form Layout**.
6. In the Property Inspector, under the **Common** tab, set the **Label** to **Modified after**.
7. In the Application Navigator, under **portal.backing**, double-click **AdvancedSearch.java** to open it.
8. Add the following import declarations. These declarations are required for the `getPredicates` method, which will be added in the next step.

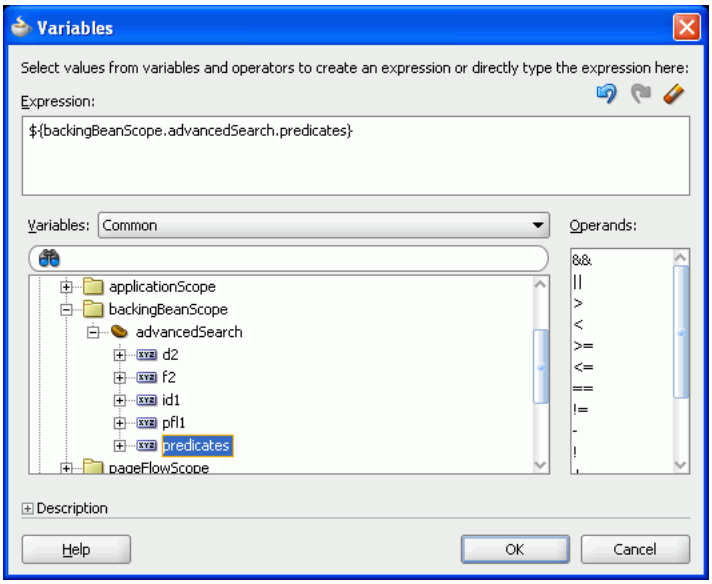
```
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.List;
import oracle.vcr.datacontrol.search.Predicate;
import oracle.vcr.datacontrol.search.Operator;
```

9. Add the following method to enable the advanced search based on last modified dates of the items stored in the repository:

```
public List<Predicate> getPredicates() {
        ArrayList<Predicate> predicates = new ArrayList<Predicate>();
        if (id1.getValue() != null && !id1.getValue().equals(""))
{
            Calendar cal = Calendar.getInstance();
            cal.setTime((Date)id1.getValue());
            predicates.add(new Predicate("jcr:content/jcr:lastModified"
                                         , Operator.GREATER_THAN
                                         , cal));
        }
        // ... other predicates
        if (predicates.size()>0)
            return predicates;
        return null;
    }
```

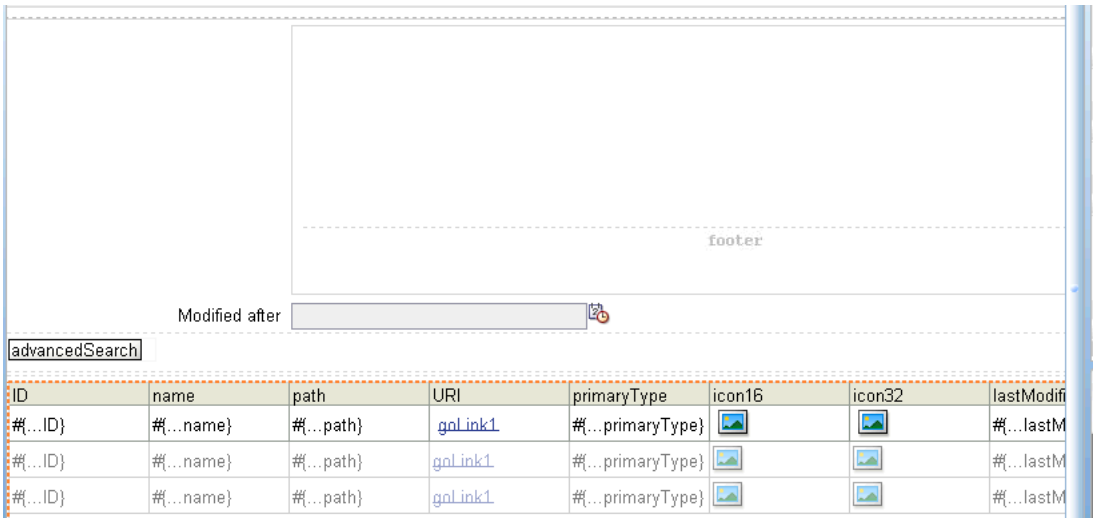
10. From the Data Controls panel, drag the **advancedSearch** node onto the page. From the Create menu, choose **Methods** and then **ADF Button**. The Edit Action Binding dialog displays.
11. For **Path**, specify the path to the directory in which the search will be performed; for example, `/`. For **isRecursive** specify `true`, and **matchAny** specify `false`.
12. For predicates, select the arrow next to the **Value** field and then select **Show El Expression Builder**. The Variables dialog displays.
13. Expand **ADF Managed Beans**, **backingBeanScope**, **advancedSearch** and select **predicates**. This adds the expression `${backingBeanScope.advancedSearch.predicates}`, as shown in Figure 28–40, and click **OK**.

Figure 28–40 Variables Dialog - predicates



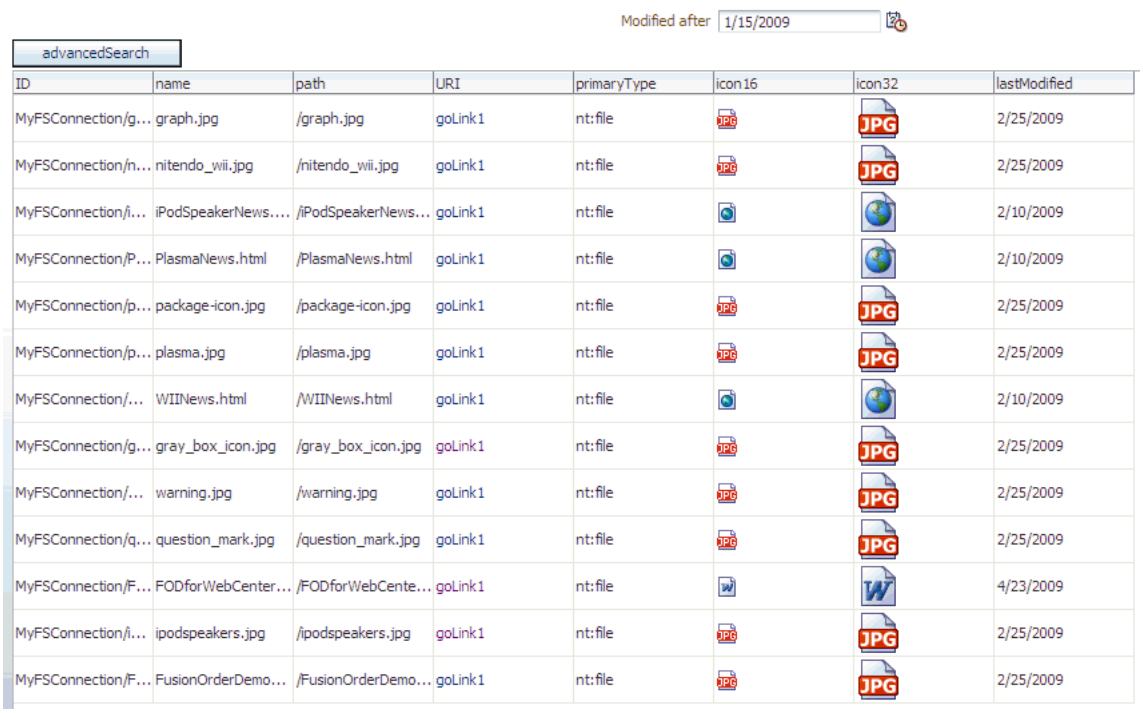
14. Click **OK** in the Edit Action Binding dialog.
15. In the Data Controls panel, expand the **advancedSearch** node. Then drag **Return** and drop it onto the page, after the advancedSearch ADF button. From the Create menu, choose **Table** and then **ADF Read-only Table**.
16. In the Edit Table Columns dialog, edit labels of the columns, if needed. Then click **OK**. The Design view looks like Figure 28–41.

Figure 28–41 Advanced Search - Design View



17. In the Property Inspector, under the **Behavior** tab, set the `ContentDelivery` property to `immediate`.
18. To view the page in a browser, under the Application Navigator, right-click the page and choose **Run**. The `advancedSearch` page displays in the browser.
19. Enter a *last modified* date and click the **advancedSearch** button. The files modified after that date are displayed, as shown in Figure 28–42.

Figure 28–42 Advanced Search Results



ID	name	path	URI	primaryType	icon16	icon32	lastModified
MyFSConnection/g...	graph.jpg	/graph.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/n...	nitendo_wii.jpg	/nitendo_wii.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/i...	iPodSpeakerNews....	/iPodSpeakerNews...	goLink1	nt:file			2/10/2009
MyFSConnection/P...	PlasmaNews.html	/PlasmaNews.html	goLink1	nt:file			2/10/2009
MyFSConnection/p...	package-icon.jpg	/package-icon.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/p...	plasma.jpg	/plasma.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/...	WIINews.html	/WIINews.html	goLink1	nt:file			2/10/2009
MyFSConnection/g...	gray_box_icon.jpg	/gray_box_icon.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/...	warning.jpg	/warning.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/q...	question_mark.jpg	/question_mark.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/F...	FODforWebCenter...	/FODforWebCente...	goLink1	nt:file			4/23/2009
MyFSConnection/i...	ipodspeakers.jpg	/ipodspeakers.jpg	goLink1	nt:file			2/25/2009
MyFSConnection/F...	FusionOrderDemo...	/FusionOrderDemo...	goLink1	nt:file			2/25/2009

28.5.8 What Happens at Runtime

Clicking the **search** button at runtime invokes the `search` method and the values provided through the UI are used as the parameters for the search. At runtime, you can perform the wild (`%`) card search. For example, to search for files that have `.jpg` extension, enter `/` in the **search_path** field, enter `%jpg` in the namePattern field and then click **Search**. All files with the `.jpg` extension will display in the read-only table.

At runtime the JCR data control uses the given parameters to construct an `XPath` query for the given parameters. For the simple search example, the query is:

```
Non-recursive:
XPath query /jcr:root/element(*, nt:file)[jcr:like(ojcr:local-name(), '%jpg%')]

Recursive:
XPath query /jcr:root//element(*, nt:file)[jcr:like(ojcr:local-name(), '%jpg%')]
```

In the advanced search example, first a backing bean is added to the page, because the backing bean is required to construct the `predicate` parameter value of the `advancedSearch` method. Then the **Panel Form Layout UI** component is dropped onto the page. In this component, the `InputDate` component is dropped, which is used at runtime to supply the date-based search criterion. The `advancedSearch` method predicates parameter allows for a combination of predicates to be supplied to the `search` method. Each can specify the value of an item's properties that must apply for the search. In this example, only a modification date is tested in the predicates, but potentially multiple tests could be included, for example a modification date and a `mimeType`.

The backing bean's `getPredicates` method is hardcoded to construct the `predicates` method from the date provided by the page at runtime. At design time, the return value of the `predicates` method is bound into the predicates `advancedSearch` method parameter. At runtime this invokes the `getPredicates`

method before invoking the `advancedSearch` method to construct the correct `predicate` value.

At runtime the JCR data control uses the given parameters to construct an `XPath` query for the given parameters. For this example the query is:

```
/jcr:root//element(*, nt:hierarchyNode)[jcr:content/@jcr:lastModified >
xs:dateTime('2009-02-15T00:00:00.000+05:30')]
```

28.5.9 What You May Need to Know When Using Search Capabilities

Consider the following points while adding search capabilities:

- How certain operations work depends on the implementation of the adapter and the underlying repository. While read and query operations are similar, full text search works differently. Another example is, the file system and Oracle Portal adapters do not support search based on the `primaryType` attribute. The only supported way to search based on type is through the element (`*`, type) construct.
- If you use the Oracle Portal adapter, then the behavior of search functionality varies depending on whether Oracle Text is enabled or not. If Oracle Text is disabled, then the search is performed in the Oracle Portal content metadata. With Oracle Text turned on, all indexed content is searched, which includes the contents of files. This also applies to Content Server. That is, Content Server-based adapter performs full text index operation on its documents using Oracle Text.
- The Oracle Portal adapter does not support translations and only returns content in the base language of a page group. Searching across multiple page groups with different base languages is not supported.

29

Creating Content Presenter Display Templates

The Content Presenter task flow allows authorized users to retrieve content from Content Server and display it in a WebCenter Portal application in real time, without assistance from an IT Department or software developers. For example, you might want to display a list of the most recent Press Releases so users can browse them and click one to read it. You can place images (a photograph or a chart, for example) or add textual content to the portal.

> **Note:** Content Presenter can only be used with Content Server based content. No other repository connection types are supported.

Content Presenter renders content using a selected *display template*. The primary focus of this chapter is creating *Content Presenter display templates*. You can create a Content Presenter template to satisfy your particular content rendering requirements or to handle your own custom content properties.

This chapter includes these topics:

- Section 29.1, "Adding Content Presenter to an Application"
- Section 29.2, "What Is a Content Presenter Display Template?"
- Section 29.3, "Using the Out-of-the-Box Display Templates"
- Section 29.4, "Creating Content Presenter Display Templates"
- Section 29.5, "Using Content Presenter Display Templates in a Framework Application"
- Section 29.6, "What Happens at Runtime?"
- Section 29.7, "Performance Considerations for Content Presenter Display Templates"

29.1 Adding Content Presenter to an Application

You can add a Content Presenter task flow at design time or runtime. For details on adding Content Presenter at design time (using JDev), see Chapter 31, "Adding Content Task Flows and Document Components to a Portal Page." For details on adding and using Content Presenter at runtime, see "Publishing Content Using Content Presenter" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

29.2 What Is a Content Presenter Display Template?

A *Content Presenter display template* is a JSFF file (JSF page fragment) that defines how Content Presenter renders content items on a portal page. WebCenter provides several out-of-the-box display templates to get you started, and you can create your own templates.

> **Tip:** You can find sample display templates in `$ORACLE_HOME/jdeveloper/webcenter/samples/contentpresenter`. These sample files were installed as part of WebCenter Portal's extension for Oracle JDeveloper.

Display templates can use the full set of Rich ADF components, so you can quickly and easily create robust and attractive templates to display your content. Note, however, that you are not required to use these components in your template.

A Content Presenter display template can handle either single content items, multiple content items, or combinations of the two. For example, a multiple content item template might render tabs for each item and then call a single item template to render the details of a selected item.

Each content item is associated with a specific *content type* defined in the Content Server repository. A content type defines the properties of the content item. Content types can map to Content Server profile definitions and Site Studio region definitions. Types are created on the Content Server. As a Content Presenter display template developer, you need to know the names of the properties defined for the associated content type so that you can define how to display the selected content item(s) on the page.

> **Tip:** One way to determine the properties for the existing content types defined in Content Server is to use the Content Presenter Configuration dialog in WebCenter Portal: Spaces. For a detailed description of this technique, see Section 29.4.5, "Discovering Content Type Property Names."

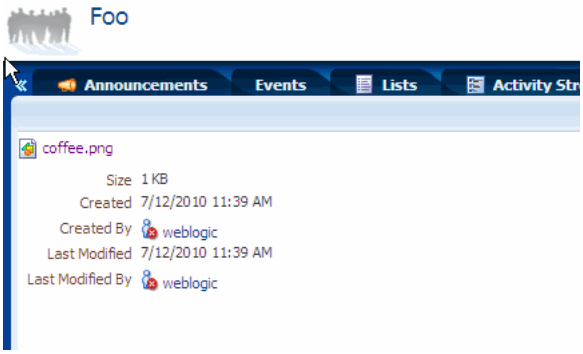
At runtime, an authorized end user can choose a display template in the Content Presenter Configuration dialog.

29.3 Using the Out-of-the-Box Display Templates

WebCenter Portal provides several out-of-the-box Content Presenter display templates. These pre-built templates provide options for displaying single content items and multiple content items.

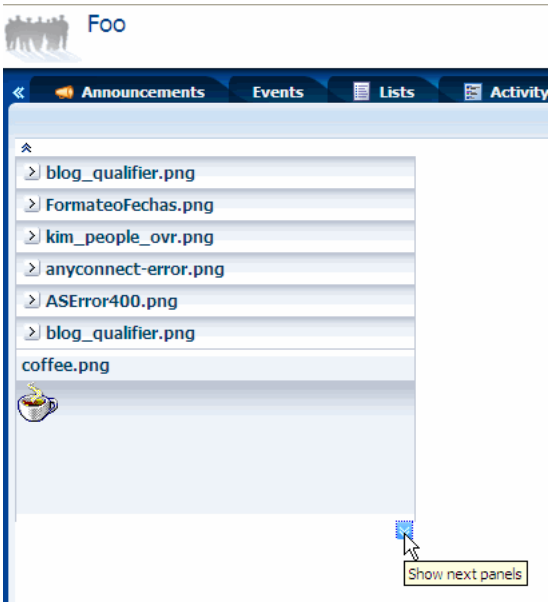
For example, the Default Document Details View template displays detailed information about any single content item including creation date, modification date, created by username, modified by username, path and any comments. Figure 29.1 shows a content item displayed at runtime with this template.

Figure 29–1 Default Document Details View Display Template



Several options are provided for displaying multiple content items, such as the Accordion View, which Displays multiple content items in an accordion format. In this format, each item can be expanded to display its details, as shown in Figure 29–2.

Figure 29–2 Accordion View Display Template



For a complete list of out-of-the-box display templates, see Section 31.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates."

> **Tip:** If you intend to create a Content Presenter display template, a good practice is to copy one of the out-of-the-box templates and modify it.

29.4 Creating Content Presenter Display Templates

If the out-of-the-box display templates (see Section 29.3, "Using the Out-of-the-Box Display Templates") do not meet your needs, you can define custom Content Presenter templates.

This section discusses these topics:

- Section 29.4.1, "Overview"

- Section 29.4.2, "Defining Single-Item Display Templates"
- Section 29.4.3, "Defining Multiple-Item Display Templates"
- Section 29.4.4, "Using EL Expressions to Retrieve Content Item Information"
- Section 29.4.5, "Discovering Content Type Property Names"

29.4.1 Overview

Depending on your needs, the approach you take to defining Content Presenter display templates will vary. Typically, you define display templates for specific single items of content, then define a multiple content item display template that includes calls to the single item display templates.

Template definitions can include calls to other display templates in any of the following ways:

- A single content item display template can call another single content item display template.
- A multiple content item display template can call a single content item display template (as shown in the examples below).
- A multiple content item display template can call another multiple content item display template.

The basic tasks for creating Content Presenter display templates include:

- Deciding whether to create a single or multiple item display template. See Section 29.4.2, "Defining Single-Item Display Templates" and Section 29.4.3, "Defining Multiple-Item Display Templates."
- Deciding what kind of information you want to display with content items. WebCenter Portal provides a set of EL expressions designed to retrieve and display this information. See Appendix G, "Expression Language Expressions" and Section 29.4.4, "Using EL Expressions to Retrieve Content Item Information."
- Designing the look and feel of the template. In addition to standard JSFF constructs, display templates can use ADF components. See *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.
- Exporting the template as a portal resource. See Section 29.5.1, "Export a Content Presenter Display Template as a Portal Resource."

29.4.2 Defining Single-Item Display Templates

Examples of single content items for which you may want to create Content Presenter display templates are:

- Individual items to display with a specific look and feel on a page.
- Different views of a specific type of item (such as short and detailed views of an article).
- Different versions of a similar item (such as Press Releases that may be formatted differently for different groups, using a different set of properties).

The definition of a single content item display template uses the JSP tags listed in Table 29–1.

Table 29–1 Content Display Template Tags for Single Content Items

JSP Tag	Description	Example

`contentTemplateDef`	Required. Defines a single content item template.
	Attributes:
	`var` - Specifies the content node that will be rendered by this display template. In the template definition code, you can use the EL expressions described in Section 29.4.4.1, "Retrieving Basic Information About a Content Item" to retrieve required information about the node.
		`<dt:contentTemplateDef var="node">`
		`  <af:outputText value="#{node.name}" />`
		`</dt:contentTemplateDef>`

`renderProperty`	Optional. Retrieves and renders the string value(s) of the specified node property inline.
	Attributes:
	`id` - Identifies this component. This value must be unique within the closest parent component that is a naming container.
	`name` - Specifies the name of the property to get. If this is a system property (`cm_createdBy`, `cm_createdDate`, `cm_modifiedDate`, and `cm_path`) then the value on the node will be used. If not specified, then the `primaryProperty` will be used if defined.
	`node` - Specifies the node to use. This value can either be a bound attribute to a managed bean, or a request attribute variable.
	`blockSize` - Specifies the size of the blocks in bytes, to read the bytes of a binary property. Default is 2048 bytes.
	`startIndex` - Specifies the index (from 0) in the document's bytes at which to start printing. Defaults to 0.
	`endIndex` - Specifies the index (from 0) in the document's bytes at which to stop printing. Defaults to value of `blockSize`.
	`rendered` - Specifies whether or not this component should be rendered (during Render Response Phase), or processed on any subsequent form submit. The default value is `true`.
		`<!--`
		`  Handling of text-based primary`
		`  properties (HTML, text, etc.).`
		`-->`
		`<dt:contentTemplateDef var="node">`
		`  <cmf:renderProperty id="rp1"`
		`  name="#{node.primaryProperty.name}"`
		`  node="#{node}"/>`
		`</dt:contentTemplateDef>`

Table 29–1 (Cont.) Content Display Template Tags for Single Content Items

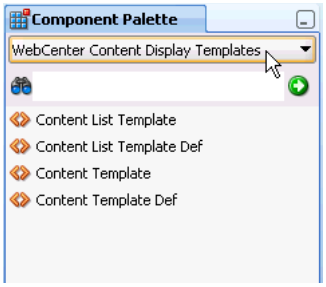
JSP Tag	Description	Example

`contentTemplate`	Optional. Nested under `contentTemplateDef`.
	Calls another single item template.
	Attributes:
	`node` - Required. Specifies the content repository node that should be displayed.
	`nodesHint` - Optional. When the display template is called in an iterating component, allows the pre-inclusion of all possible templates. This attribute is not needed when `contentTemplate` is used inside a `contentTemplateDef` tag.
	`view` - Optional. Specifies the target view name.
	`id` - Optional. Specifies the JSF component ID.
	`rendered` - Optional. Specifies whether the component should be rendered.
		`<dt:contentTemplateDef var="node">`
		`<af:outputText value="#{node.name}" >`
		`<dt:contentTemplate node="#{node}"`
		`view="templates.pressrelease.item"`
		`/>`
		`</af:outputText>`
		`</dt:contentTemplateDef>`

To define a display template for a single content item:

1. In JDeveloper, create a JSFF file:
   a. From the **File** menu, choose **New**.
   b. In the New Gallery dialog, expand **Web Tier**, select **JSF**, then **JSF Page Fragment**.
   c. Click **OK**.
   d. In the Create New JSF Page Fragment dialog, enter a name for the display template file.
   e. Set the Directory field to point to the oracle/webcenter/portalapp/pages folder of your portal web project. This directory must be set from the correct root path. In JDeveloper, the root path is *PROJECT_NAME* > Web Content. On the file system, the root path is *PROJECT_ROOT*/public_html. For example: *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/pages. A good practice is to first create a subfolder in `pages` and store the JSFF in the subfolder.
2. Select **View**, then **Component Palette** to open the Component Palette.
3. From the dropdown list at the top of the Component Palette (Figure 29–3):
   - Select **WebCenter Content Display Templates** for a list of display template tags.
   - Select **WebCenter Content Management Faces** for the `renderProperty` tag.

Figure 29–3 Content Display Template Tags in Component Palette



4. In Source view, drag and drop the required display template tags from the Component Palette onto the page to define your template. Refer to Table 29–1 for information about each tag and required parameter values.

Example 29–1 and Example 29–2 show sample single content item display template definitions. These examples illustrate a use case where a certain kind of document (a press release) is produced by two different departments inside a company, and each department has defined its own content type and properties. These sample Content Presenter display templates allow these two different content types to be displayed in a consistent manner.

The template shown in Example 29–1 handles a Press Release that uses a property named `xHeading` to describe the heading of the Press Release document, and `xDestinationUrl` to describe the location of the document. To learn how you can obtain these property names for a given content type, see Section 29.4.5, "Discovering Content Type Property Names."

Example 29–1 Sample Content Presenter Display Template for a Press Release Document

```
<?xml version = '1.0'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
          version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:dt="http://xmlns.oracle.com/webcenter/content/templates">
   <dt:contentTemplateDef var="node">
      <af:goImageLink text="#{node.propertyMap['xHeading'].value}"
                      id="gil1"
                      icon="#{node.icon.smallIcon}"
                      destination="#{node.propertyMap['xDestinationUrl'].value}"
                      targetFrame="_blank">
      </af:goImageLink>
   </dt:contentTemplateDef>
</jsp:root>
```

The template shown in Example 29–2 handles a Press Release that uses a property named `dDocTitle` to describe the heading of the Press Release document, and `xLinkUrl` to describe the location of the document.

Example 29–2 Sample Content Presenter Template for Another Press Release Document

```
<?xml version = '1.0'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
          version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:dt="http://xmlns.oracle.com/webcenter/content/templates">
```

```
   <dt:contentTemplateDef var="node">
      <af:goImageLink text="#{node.propertyMap['dDocTitle'].value}"
                      id="gil1"
                      icon="#{node.icon.smallIcon}"
                      destination="#{node.propertyMap['xLinkUrl'].value}"
                      targetFrame="_blank">
      </af:goImageLink>
   </dt:contentTemplateDef>
</jsp:root>
```

29.4.3 Defining Multiple-Item Display Templates

Examples of multiple content items for which you may want to create Content Presenter display templates are:

- A group of similar items that you want to display on a page, such as a list of books or an employee directory of pictures.
- Query results, such as all documents modified in the last week.
- A list of all documents in a Content Server folder.

The definition of a multiple content item display template uses the JSP tags listed in Table 29–2.

Table 29–2 Content Display Template Tags for Multiple Content Items

JSP Tag	Description	Example

`contentListTemplateDef`	Required. Defines the multiple content item template.
	
	Attributes:
	
	`var` - Specifies the content node that will be rendered by this display template. In the template definition code, you can use the EL expressions described in Section 29.4.4.1, "Retrieving Basic Information About a Content Item" to retrieve required information about the node.
		`<dt:contentListTemplateDef var="nodes">`
		`  <af:iterator value="#{nodes}" var="node">`
		`  <af:outputText value="#{node.name}" />`
		`  </af:iterator>`
		`</dt:contentListTemplateDef>`

Table 29–2 (Cont.) Content Display Template Tags for Multiple Content Items

JSP Tag	Description	Example

`contentListTemplate`	Optional. Nested under `contentListTemplateDef`.
	Calls another multiple item template.
	Attributes:
	`nodes` - Required. Provides the list of VCR nodes that should be displayed
	`category` - Required. Specifies the target template category.
	`view` - Required. Specifies the target view name.
	`id` - Optional. Specifies the JSF component ID.
	`rendered` - Optional. Specifies whether the component should be rendered.
		`<dt:contentListTemplateDef var="nodes">`
		`<!--`
		`Reuse the default bulleted list view, but indent it with a <blockquote>`
		`-->`
		`<f:verbatim>`
		`<blockquote>`
		`</f:verbatim>`
		`<dt:contentListTemplate nodes="#{nodes}"`
		`category="oracle.webcenter.content.templates.default.category"`
		`view="oracle.webcenter.content.templates.default.list.bulleted"/>`
		`<f:verbatim>`
		`</blockquote>`
		`</f:verbatim>`
		`</dt:contentListTemplateDef>`

`contentTemplate`	Optional. Nested under `contentListTemplateDef`.
	Calls a single item template.
	Attributes:
	`node` - Required. Specifies the content repository node that should be displayed.
	`nodesHint` - Optional. When the display template is called in an iterating component, allows the pre-inclusion of all possible templates. This attribute is usually required when `contentTemplate` is used inside a `contentListTemplateDef` tag.
	`view` - Optional. Specifies the target view name.
	`id` - Optional. Specifies the JSF component ID.
	`rendered` - Optional. Specifies whether the component should be rendered.
		`<dt:contentListTemplateDef var="nodes">`
		`<af:iterator rows="0" var="node"`
		`varStatus="iterator" value="#{nodes}">`
		`<dt:contentTemplate node="#{node}"`
		`view="templates.pressrelease.listitem"`
		`nodesHint="#{nodes}"/>`
		`</af:iterator>`
		`</dt:contentListTemplateDef>`

To define a display template for multiple content items:

1. In JDeveloper, create a JSFF file and open the Component Palette, as described in Step 1 through Step 3 in Section 29.4.2, "Defining Single-Item Display Templates".
2. In Source view, drag and drop the required display template tags from the Component Palette onto the page. Refer to Table 29–2 for information about each tag and required parameter values.

Example 29–3 shows a sample multiple content item display template definition.

Example 29–3 Sample display template definition for the display of multiple Press Releases (press-release-list-view.jsff)

This template definition iterates over the data items selected for display, and processes them according to the referenced view (`view="mycorp.content.templates.pressrelease.listitem"`).

```
<?xml version = '1.0'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:dt="http://xmlns.oracle.com/webcenter/content/templates">
   <dt:contentListTemplateDef var="nodes">
      <af:panelGroupLayout layout="scroll" id="nodeListPanel" valign="middle">
        <af:iterator rows="0" var="node" varStatus="iterator" value="#{nodes}">
           <dt:contentTemplate node="#{node}"
               view="mycorp.content.templates.pressrelease.listitem"
               nodesHint="#{nodes}"/>
        </af:iterator>
      </af:panelGroupLayout>
   </dt:contentListTemplateDef>
</jsp:root>
```

29.4.4 Using EL Expressions to Retrieve Content Item Information

This section describes the EL expressions that you can use in your Content Presenter display template definitions to retrieve and display specific information about content items.

Use the EL expressions described in the following tables as you define your Content Presenter display templates as explained in Section 29.4.2, "Defining Single-Item Display Templates" and Section 29.4.3, "Defining Multiple-Item Display Templates." These expressions are used with the JSP tags described in Table 29–1 and Table 29–2.

29.4.4.1 Retrieving Basic Information About a Content Item

The EL expressions listed in Table 29–3 allow you to display basic information about a content item in a display template.

Table 29–3 EL Expressions for Retrieving Basic Content Information

EL Expression	Description
`#{node.createdBy}`	Returns the user name of the node's creator.
`#{node.createdDate}`	Returns the node's creation date.
`#{node.hasParentNode}`	Returns `true` if the current node has a valid parent node ID, or a node that has no parent.
`#{node.icon}`	Returns the icon service defined in the current web application.
`#{node.id}`	Returns the node's identifier.
`#{node.isFolder}`	Returns `true` if this node is associated with a folder or container.
`#{node.isInherited}`	Returns `true` if this node is inherited by another object class definition.
`#{node.modifiedBy}`	Returns the user name of the node's last modifier.
`#{node.modifiedDate}`	Returns the node's last modification date.
`#{node.name}`	Returns the node's name.

Table 29–3 (Cont.) EL Expressions for Retrieving Basic Content Information

EL Expression	Description
`#{node.parentId}`	Returns the parent node's identifier.
`#{node.path}`	Returns the node's path.
`#{node.primaryProperty}`	Returns the node's primary property, if available.
`#{node.propertyMap}`	Creates and returns a map of wrapped property objects, keyed by property name. Properties can be accessed as `#{node.propertyMap['myProp']}` or `#{node.propertyMap.myProp}`.
`#{node.url}`	Returns an instance of the node property URL service for the primary property of this node (if any). By default, resolves to `#{node.url.renderUrl}`. This is a shortcut for `#{node.primaryProperty.url}`.

29.4.4.2 Working with Content Item Properties and Values

Use the EL expression described in Table 29–4 and Table 29–5 to perform actions on content item node properties and property values.

Tip: To determine the names of the properties defined for a given content type, see Section 29.4.5, "Discovering Content Type Property Names."

Table 29–4 EL Expressions for Content Item Node Properties

EL Expression	Description
`#{node.propertyMap['myProp']'myProp'].asTextHtml}`	Returns this property as text or HTML if the type is text or HTML. If `isHTML` or `isPlainText` is `true`, the text or HTML is returned as a string.
`#{node.propertyMap['myProp'].hasValue}`	Returns `true` if a value is associated with this property.
`#{node.propertyMap['myProp'].icon}`	Returns the icon service defined in the current web application.
`#{node.propertyMap['myProp'].indexedName}`	Returns the indexed name of a multi-valued property. For example, if a multi-valued node property named `color` contains `blue`, `red`, `orange`, the indexed name of the `red` value is `color[1]`. The following EL expression references the color `orange` in this list: `#{node.propertyMap['color[2]'].value.stringValue}`
`#{node.propertyMap['myProp'].isAudio}`	Returns `true` if the property's mime type is 'audio/'.
`#{node.propertyMap['myProp'].isBinary}`	Returns `true` if the current property is of type `Property.BINARY`.
`#{node.propertyMap['myProp'].isBoolean}`	Returns `true` if the current property is of type `Property.BOOLEAN`.
`#{node.propertyMap['myProp'].isCalendar}`	Returns `true` if the current property is of type `Property.CALENDAR`.
`#{node.propertyMap['myProp'].isDouble}`	Returns `true` if the current property is of type `Property.DOUBLE`.
`#{node.propertyMap['myProp'].isExcel}`	Returns `true` if the property's mime type is 'application/vnd.ms-excel', 'application/excel', 'application/x-excel', or 'application/x-msexcel'.

Table 29–4 (Cont.) EL Expressions for Content Item Node Properties

EL Expression	Description
`#{node.propertyMap['myProp'].isGIF}`	Returns `true` if the property's mime type is '`image/gif`'.
`#{node.propertyMap['myProp'].isHTML}`	Returns `true` if the property's mime type is '`text/html`'.
`#{node.propertyMap['myProp'].isImage}`	Returns `true` if the property's mime type is '`image/`'.
`#{node.propertyMap['myProp'].isJPEG}`	Returns `true` if the property's mime type is '`image/jpeg`'.
`#{node.propertyMap['myProp'].isLink}`	Returns `true` if the current property is of type `Property.LINK`.
`#{node.propertyMap['myProp'].isLong}`	Returns `true` if the current property is of type `Property.LONG`.
`#{node.propertyMap['myProp'].isMSWord}`	Returns `true` if the property's mime type is '`application/vnd.ms-word`' or '`application/msword`'.
`#{node.propertyMap['myProp'].isMultiValued}`	Returns `true` if this property is multi-valued.
`#{node.propertyMap['myProp'].isNested}`	Returns `true` if the current property is of type `Property.NESTED`.
`#{node.propertyMap['myProp'].isPDF}`	Returns `true` if the property's mime type is '`application/pdf`'.
`#{node.propertyMap['myProp'].isPlainText}`	Returns `true` if the property's mime type is '`text/plain`'.
`#{node.propertyMap['myProp'].isPNG}`	Returns `true` if the property's mime type is '`image/png`'.
`#{node.propertyMap['myProp'].isPowerPoint}`	Returns `true` if the property's mime type is '`application/vnd.ms-powerpoint`', '`application/mspowerpoint`', or '`application/x-mspowerpoint`'.
`#{node.propertyMap['myProp'].isPrimaryProperty}`	Returns `true` if this property is the primary property.
`#{node.propertyMap['myProp'].isRequired}`	Returns `true` if this property is required/mandatory.
`#{node.propertyMap['myProp'].isRestricted}`	Returns `true` if this property is restricted.
`#{node.propertyMap['myProp'].isRichText}`	Returns `true` if the property's mime type is '`text/richtext`'.
`#{node.propertyMap['myProp'].isString}`	Returns `true` if the current property is of type `Property.STRING`.
`#{node.propertyMap['myProp'].isTextBased}`	Returns `true` if this property is text-based (`isHTML`, `isPlainText`, `isString`, `isCalendar`, `isBoolean`, `isDouble`, `isLong`).
`#{node.propertyMap['myProp'].isVideo}`	Returns `true` if the property's mime type is '`video/`'.
`#{node.propertyMap['myProp'].isXML}`	Returns `true` if the property's mime type is '`text/xml`'.
`#{node.propertyMap['myProp'].isZip}`	Returns `true` if the property's mime type is '`application/zip`'.
`#{node.propertyMap['myRefNode'].linkAsNode}`	Returns the `myRefNode` property as a node, where `myRefNode` is a link property type. Properties can be referenced in EL expressions. **Example**: `#{node.propertyMap['myRefNode'].linkAsNode.primaryProperty.url.renderUrl}`

Table 29–4 (Cont.) EL Expressions for Content Item Node Properties

EL Expression	Description
`#{node.propertyMap['myProp'].name}`	Returns the property's name.
`#{node.propertyMap['myProp'].nestedProperty}`	Retrieves nested properties for this single-valued property, and returns a list of properties.
`#{node.propertyMap['myProp'].type}`	Returns the data type of this property value. For example: `String`, `Integer`, `Long`, and so on.
`#{node.propertyMap['myProp'].url}`	Returns a URL service for this property.
`#{node.propertyMap['myProp'].value}`	Returns the value service for this property.
#{node.propertyMap['myProp'].nestedProperties}	Returns elements from a static list. For example: `<af:iterator var="listItem" value="#{node.propertyMap['ARTICLE_RGD:Paragraphs'].nestedProperties}" varStatus="vs"> <af:outputText id="ot1" value='#{listItem[0].value}'/> <af:outputText id="ot3" value="#{listItem[1].value}"/> </af:iterator>`
`#node.propertyMap['regionDefName:elementName'].asTextHtml}`	Returns Site Studio data as HTML text. For example: `#node.propertyMap['RD_NEWS:LEAD'].asTextHtml}` The element name (`LEAD`) is prefixed by the Region Definition name (`RD_NEWS`). See also Section 29.4.7, "Referencing Site Studio Region Elements in a Custom View."

Table 29–5 EL Expressions for Content Item Node Property Values

EL Expression	Description
`#{node.propertyMap['myProp'].value.binaryValue}`	Returns custom attributes for a binary property type or attachment. Attributes available on `binaryValue` are: ■ `contentType` – Returns the mime type of the binary content (for example, `text` or `html`). ■ `name` – Returns the filename of the binary content (for example, `index.html`). ■ `size` – Returns the size of the binary content, or `-1` if the size is unavailable.
`#{node.propertyMap['myProp'].value.booleanValue}`	Returns the value of this property as `java.lang.Boolean`.
`#{node.propertyMap['myProp'].value.calendarValue}`	Returns the value of this property as `java.util.Calendar`.
`#{node.propertyMap['myProp'].value.doubleValue}`	Returns the value of this property as `java.lang.Double`.

Table 29–5 (Cont.) EL Expressions for Content Item Node Property Values

EL Expression	Description
`#{node.propertyMap['myProp'].value.longValue}`	Returns the value of this property as `java.lang.Long`.
`#{node.propertyMap['myProp'].value.orderedPosition}`	Returns the "index" of the property when the property is multi-valued. **Example**: `#{node.propertyMap['address[0]'].value.orderedPosition},`
`#{node.propertyMap['myProp'].value.stringValue}`	Returns the value of this property as `java.lang.String`. **Example**: `#{node.propertyMap['firstName'].value.stringValue}`

29.4.4.3 Working with Content Item Icons and URLs

Table 29–6 and Table 29–7 describe EL expressions for working with icons and URLs associated with content items and properties.

Table 29–6 EL Expressions for Content Item Node or Property Icons

EL Expression	Description
`#{node.icon.largeIcon}`	Returns a URL to an image resource for a large icon.
`#{node.propertyMap['myDoc'].icon.largeIcon}`	**Example**: `<af:image source="#{node.propertyMap['projectFolder'].largeIcon}"/>`
`#{node.icon.smallIcon}`	Returns a URL to an image resource for a small icon.
`#{node.propertyMap['myDoc'].icon.smallIcon}`	**Example**: `<af:image source="#{node.icon.smallIcon}" />`

Table 29–7 EL Expressions for Content Item Node URLs

EL Expression	Description
`#{node.url.downloadUrl}`	Creates a URL to the binary content. Forces a download, and the underlying operating system renders the content based on the content type. **Example**: `<af:goLink destination="#{node.url.downloadUrl}" targetFrame="_blank"/>`
`#{node.url.renderUrl}`	Creates a URL to the binary content. Allows the browser to render the content based on the content type. By default, `#{node.url}` resolves to `#{node.url.renderUrl}`. **Example**: `<af:goLink destination="#{node.url.renderUrl}" targetFrame="_blank"/>`

29.4.5 Discovering Content Type Property Names

This section describes a useful technique for learning the property names of a content item's content type.

Note: You can also use REST services to obtain content type property names. For more information see the *Oracle Fusion Middleware Content Management REST Service Developer's Guide*.

Each content item is associated with a specific *content type* defined in the Content Server repository. Content types can map to Content Server profile definitions and Site Studio region definitions. Types are created on the Content Server. A content type defines the properties of the content item (see Table 29–8), and is identified when you register the template.

As a Content Presenter display template developer, you will need to know the names of the properties defined for the associated content type so that you can define how to display the selected content item(s) on the page.

One way to determine the properties for the existing content types defined in Content Server is to use the Content Presenter Configuration dialog in WebCenter Portal: Spaces:

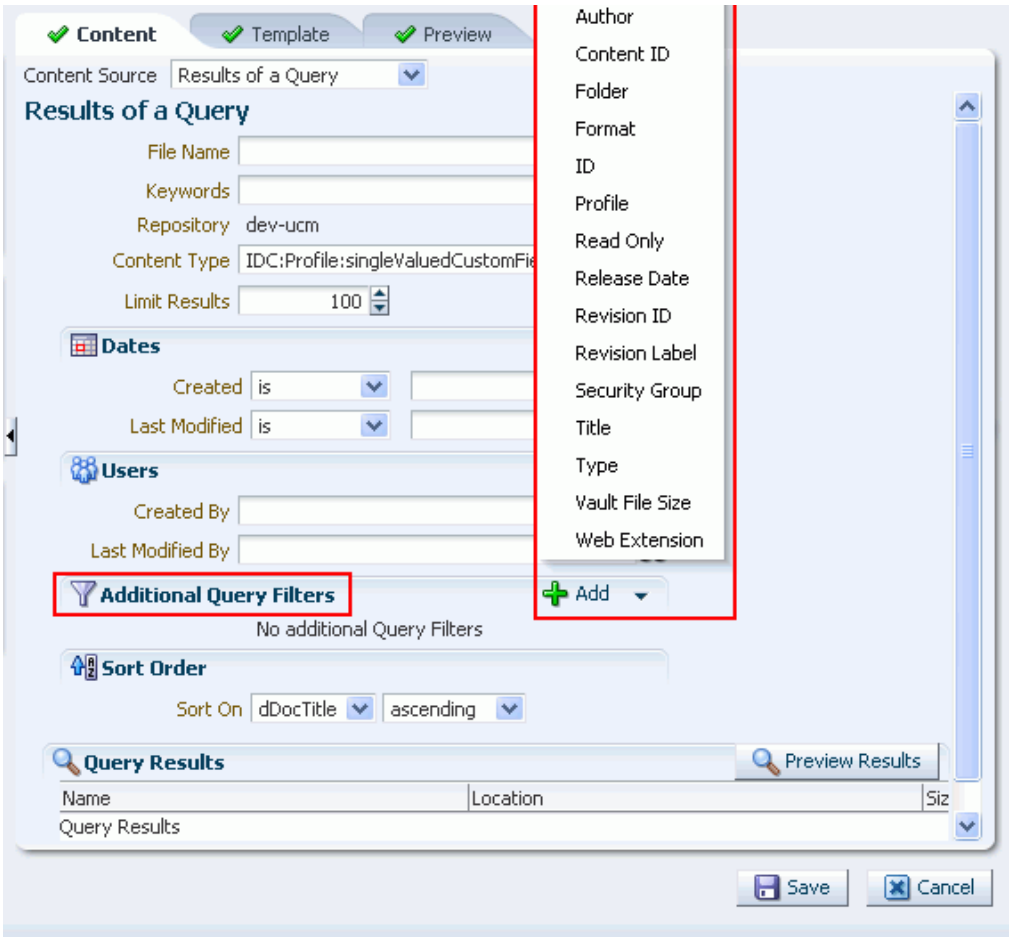
1. Open the Content Presenter Configuration dialog. For instructions, refer to the section "Displaying Content Using the Content Presenter Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
2. On the **Content** tab of the Content Presenter Configuration dialog, select **Results of a Query** to a view a list of content types.
3. Select the **Content Type** for the content item that you wish to display.
4. Next to **Additional Query Filters**, click **Add** to list all the properties that are defined for the selected content type (Figure 29–4).

Note: The list of property names shown in the dropdown list (Figure 29–4) are display names only. For example, the display name `Author` corresponds to the actual content type property name `dDocAuthor`.

To discover the actual property names that correspond to the display names, you must access them through the Content Server administration console, or contact your Content Server administrator.

When referencing the actual content type properties, you must include the leading letter `d` or `x` to reference the internal Content Server property name. The prefix `d` identifies a system property, and the prefix `x` identifies a user property. For example: `dDocAuthor` or `xCollectionID`.

Figure 29–4 Identifying Content Type Properties



29.4.6 Referencing External Files in Display Templates

In some cases, a display template needs to reference an external file, like a CSS file. All such references must be either an absolute path or a path that is relative to the root of the web application. For example:

- **absolute path** – `http://host:port/mypath/file.css`
- **relative path** – `/webcenter/mypath/file.css`

Do not use local references to external files. Local references to external files do not work because they are not included when you upload a Content Presenter display template to a WebCenter Portal: Spaces application, as explained in Section 29.5, "Using Content Presenter Display Templates in a Framework Application."

29.4.7 Referencing Site Studio Region Elements in a Custom View

You can use custom display templates to display Site Studio Region Definition elements. For example, you might have a Site Studio Region Definition called RD_NEWS with four elements: TITLE, LEAD, IMAGE, and BODY. A Content Presenter display template can reference these elements using the node property EL expression like this:

```
#node.propertyMap['RD_NEWS:LEAD'].asTextHtml}
```

Example 29–4 illustrates how these Site Studio Region elements can be included in a `contentTemplateDef` definition:

Example 29–4 Referencing Site Studio Region Elements in a Template

```
<dt:contentTemplateDef var="node">
   <af:panelGroupLayout layout="vertical" id="pgl3">
     <af:panelGroupLayout layout="horizontal" valign="top" inlineStyle="background-color:#FFF;
padding:10px;" id="pgl4">
         <af:panelGroupLayout layout="vertical" id="pgl2" valign="top">
            <af:outputText value="#{node.propertyMap['dInDate'].value.calendarValue}" id="ot3"
styleClass="bodytext" converter="javax.faces.DateTime"/>
          </af:panelGroupLayout>
          <af:spacer width="10px;" id="s1" inlineStyle="background-color:#DDD; color:white;"/>
          <af:panelGroupLayout layout="vertical" id="pgl1" valign="top">
             <af:outputText value="#{node.propertyMap['xTargetGroup'].value}" id="ot12"
inlineStyle="background-color:#0A9FC0; color:white; text-align:left; padding:5px;"/>
            <af:goLink text="#{node.propertyMap['RD_NEWS:TITLE'].asTextHtml}" id="gil1"
                destination="#{'/faces/home/news-viewer?news_
id='}#{node.propertyMap['dDocName'].value}" styleClass="newstitle"/>
           <af:outputText value="#{node.propertyMap['RD_NEWS:LEAD'].asTextHtml}" id="ot2"
styleClass="bodytext"/>
         </af:panelGroupLayout>
       <af:panelGroupLayout layout="vertical" id="pgl32" valign="top" styleClass="newsimage">
         <af:outputText value="#{node.propertyMap['RD_NEWS:IMAGE'].asTextHtml}" escape="false"
id="ot1"  inlineStyle="max-width:100px;"/>
       </af:panelGroupLayout>
     </af:panelGroupLayout>
     <af:panelGroupLayout layout="horizontal" id="aaaa">
     </af:panelGroupLayout>
   </af:panelGroupLayout>
</dt:contentTemplateDef>
```



For a complete, end-to-end example illustrating how to reference Site Studio Region elements in multiple templates, see the WebCenter Architecture Team blog entry at: `http://blogs.oracle.com/ATEAM_WEBCENTER/entry/content_presenter_cmis_complete`.

29.4.8 Using the Content Presenter Display Template

For information on using a display template in a Content Task flow, see Section 31.5, "Adding a Content Task Flow to a Page." See also Section 31.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates." For information on integrating a display template into a WebCenter Portal: Framework application, see the following section Section 29.5, "Using Content Presenter Display Templates in a Framework Application."

> **Note:** If you display a wiki page in a Framework application using a Content Presenter display template, by default any links within that wiki page are displayed in the Document Viewer. If you want to display wiki page links using Content Presenter, you must edit the `adf-config.xml` file. For more information, see Section 32.3.5, "Displaying Wiki Page Links Within Content Presenter."

29.5 Using Content Presenter Display Templates in a Framework Application

This section explains how to make a Content Presenter display template available to Content Presenter in a Framework application.

- Section 29.5.1, "Export a Content Presenter Display Template as a Portal Resource"
- Section 29.5.2, "Upload the New Content Presenter Display Template"
- Section 29.5.3, "Test the New Content Presenter Display Template"

Note: If you are using a locally configured JDev environment to develop a Framework application, all you need to do to use a Content Presenter display template in your application is to export it as a portal resource, as explained in Section 29.5.1, "Export a Content Presenter Display Template as a Portal Resource." You do not need to perform the upload task as explained in Section 29.5.2, "Upload the New Content Presenter Display Template."

29.5.1 Export a Content Presenter Display Template as a Portal Resource

This section explains how to export a Content Presenter display template as a portal resource. This procedure is a prerequisite to adding a Content Presenter display template to a deployed WebCenter Portal: Spaces application and to Content Presenter at runtime.

1. Create a Content Presenter display template, as explained previously in Section 29.4, "Creating Content Presenter Display Templates."
2. In the Application Navigator, right-click the Content Presenter display template file (a JSFF file) and select **Create Portal Resource**.
3. Fill in the Create Portal Resource dialog. This dialog differs depending on whether you are creating a single or multiple-item template. The dialog for a single-item template is shown in Figure 29–5. The dialog for a multiple-item template is shown in Figure 29–6.

Figure 29–5 Create Portal Resource Dialog for a Single-Item Template

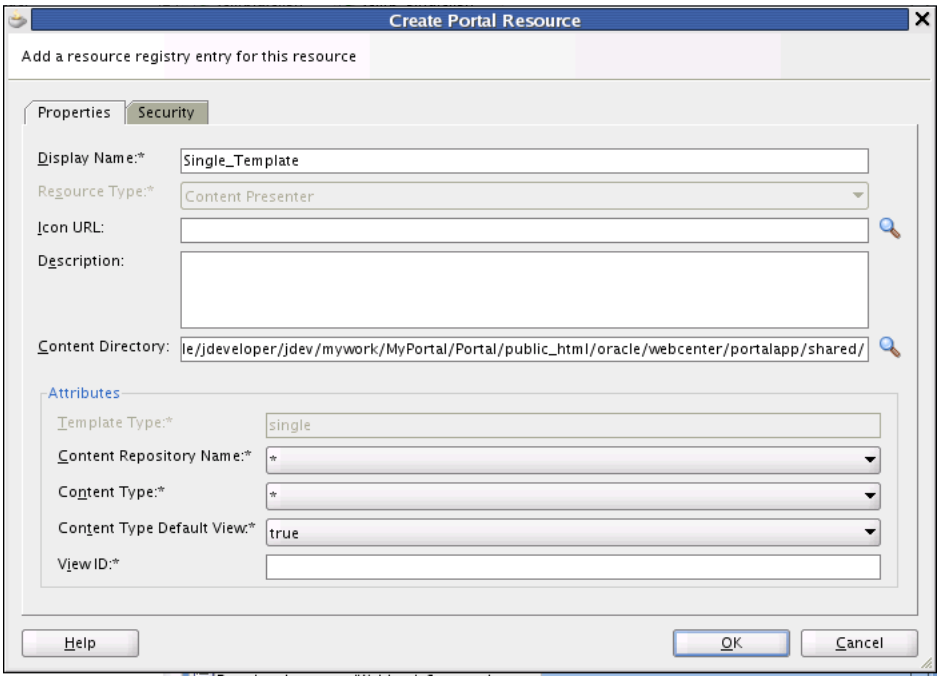
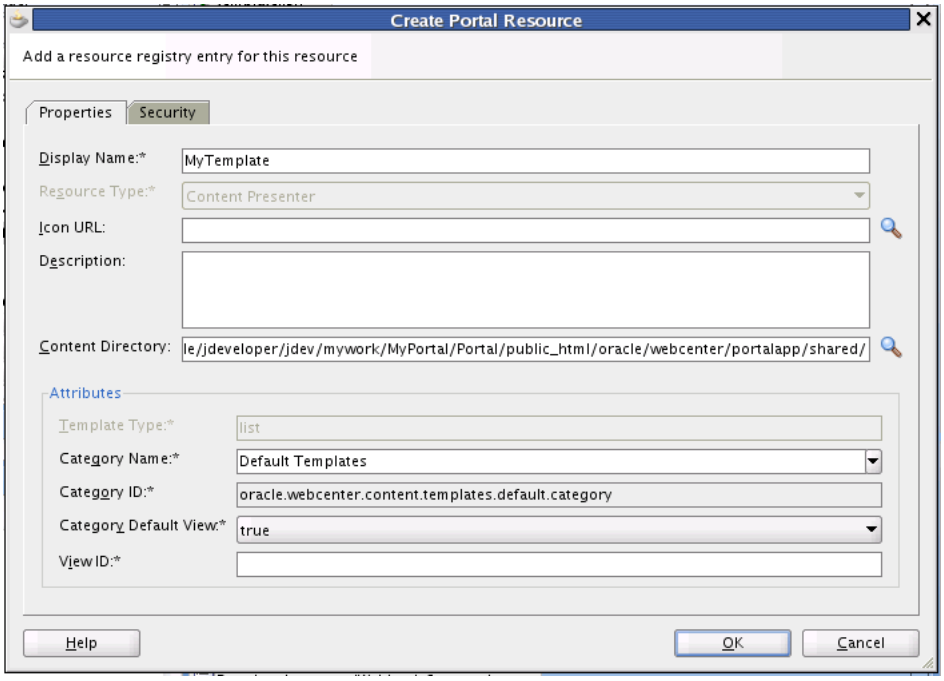


Figure 29–6 Create Portal Resource Dialog for a Multiple-Item Template



The default settings in the Create Portal Resource dialog are generally sufficient; however, you must enter a unique value in the View ID field. The Display Name is the name that appears in the display template drop down menu at runtime.

More On OTN

The View ID parameter is intended to be human-readable; therefore, it is not automatically generated for you. For example, you can use the View ID to programmatically refer to one template from another. For example, a multiple content item template could render tabs for each item and also call on a single item

template to be used below the selected tab to render the details of that item. As another example, you can include one template in another by passing the View ID as a parameter to the Content Presenter task flow. For a detailed example illustrating the latter use case, see the WebCenter Architecture Team blog entry at: `http://blogs.oracle.com/ATEAM_WEBCENTER/entry/content_presenter_cmis_complete`. See also Section 29.4.7, "Referencing Site Studio Region Elements in a Custom View."

Additional Attributes in the Create Portal Resource dialog for a **single item template** include:

- **Template Type** – Reflects whether the template is single or multi-item template.
- **Content Repository Name** – By default, the portal resource uses the repository associated with the application's default content connection. If you intend to export the portal resource for use in a WebCenter Portal: Spaces application, you must pick the repository that matches the one used by the WebCenter Portal: Spaces connection name(s).
- **Content Type** – Lets you pick a content type for which the template will be valid. The list of content types displayed in the menu depends on the selected content repository.
- **Content Type Default View** – If set to true, specifies that the display template is the default template to use for the given combination of repository and content type.
- **View ID** – A unique value that is used to refer to the template programmatically, as discussed previously in this section.

Additional Attributes in the Create Portal Resource dialog for a **multiple item template** include:

- **Template Type** – Reflects whether the template is single or multi-item template.
- **Category Name** – A name given to a category of display templates. Can be any name that describes the category or you can use the **Default Templates** category name.
- **Category ID** – An ID for the category. If you select **Default Templates** as the category name, this field will become read-only and populate with **oracle.webcenter.content.templates.default.category**. If you are creating a new category, this value can be any unique string. For example: **oracle.webcenter.content.templates.pressrelease.category**. If two resources have the same ID, they will be grouped together in the user interface.
- **View ID** – A unique value that is used to refer to the template programmatically, as discussed previously in this section.

Note: For more information on other fields in the Create Portal Resource dialog, see Section 17.4.3, "How to Export a Portal Resource from JDeveloper."

---

4. Click **OK** in the dialog when you are finished.
5. Right-click the Content Presenter display template (JSFF) file again, and this time select **Export Portal Resource**.

6. In the Export Portal Resource dialog, enter or browse to a directory in which to place the exported portal resource file. You can place this file anywhere on your file system.
7. Click **OK** in the Export Portal Resource dialog.

Now that your Content Presenter display template is exported as a portal resource, the next step is to upload it to your WebCenter Portal: Spaces application. After uploading the display template portal resource, it will be available in the list of display templates when you add a Content Presenter to your portal page.

29.5.2 Upload the New Content Presenter Display Template

You can upload a new display template to any deployed WebCenter Portal: Framework or WebCenter Portal: Spaces application that includes the Resource Manager. To upload a Content Presenter display template to a WebCenter Portal: Framework or WebCenter Portal: Spaces application, do the following:

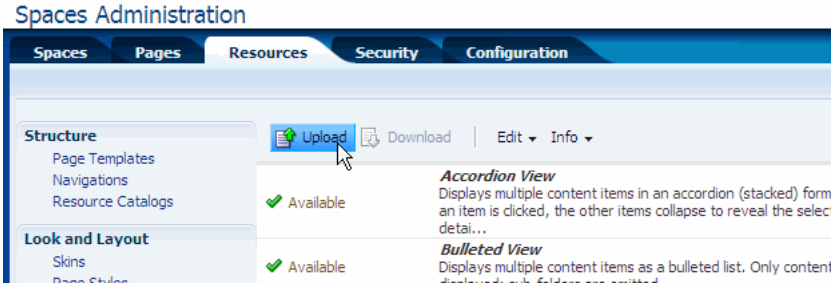
Note: The repository connection names and details must be the same between the design time Framework application in which you developed the Content Presenter display template and the WebCenter Portal: Spaces application.

Note: Any references to external files in the display template must either be absolute or relative path references. See Section 29.4.6, "Referencing External Files in Display Templates."

---

1. Start your WebCenter Portal: Framework or WebCenter Portal: Spaces application.
2. Navigate to the page to which you want to add the Content Presenter display template.
3. Click **Administration**.
4. Select the **Resources** tab.
5. Under the Look and Layout heading (along the right side of the web page), click **Content Presenter**.
6. Select **Upload**, as shown in Figure 29–7.

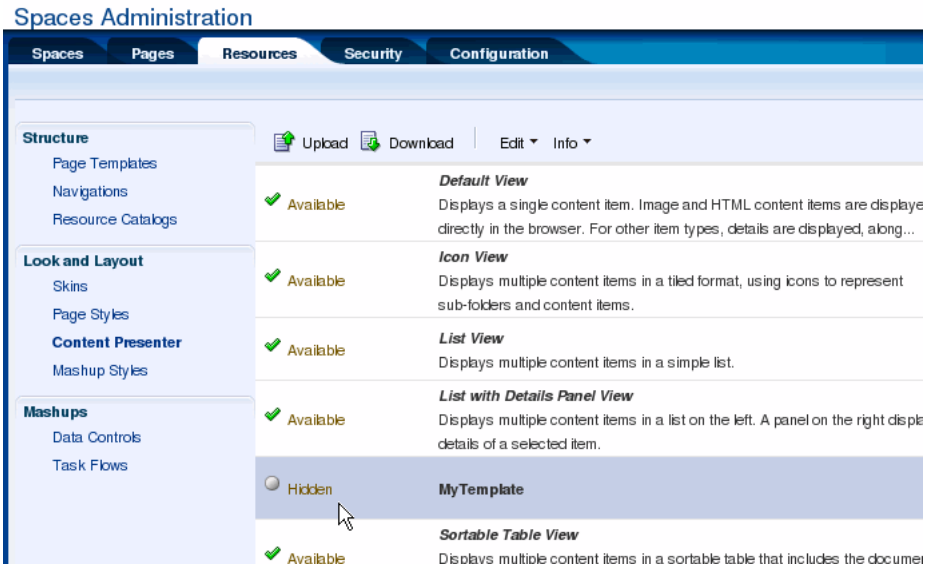
Figure 29–7 Uploading the Display Template Portal Resource



7. In the Upload dialog, navigate to the portal resource file you saved on your file system.

8. After you've selected the template file, click **OK**. If the operation is successful, an Information dialog appears with the message "Resource uploaded successfully."
9. Locate your Content Presenter display template in the list view of the Resources tab and select it from the list, as shown in Figure 29–8.

Figure 29–8 Toggling the Show/Hide State of a Template



10. If the template state is currently hidden, select **Show** from the Edit menu. If a template is hidden, it won't be available for use in Content Presenter.

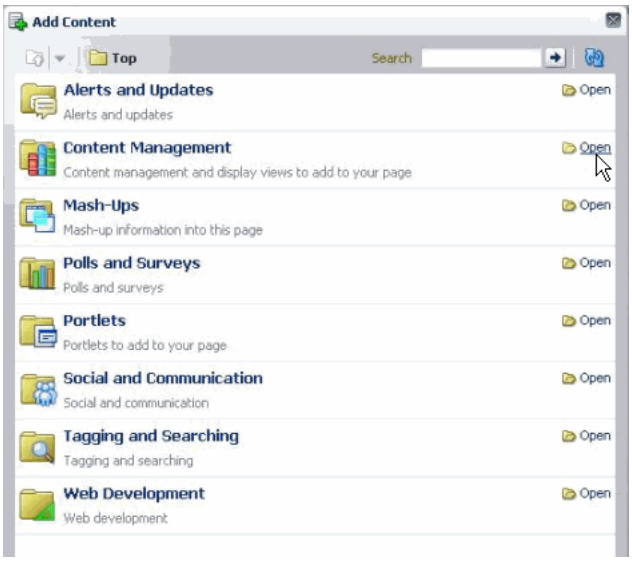
The new Content Presenter display template is now ready to be used with Content Presenter in a WebCenter Portal: Spaces application.

29.5.3 Test the New Content Presenter Display Template

This section explains how to test your Content Presenter display template with Content Presenter.

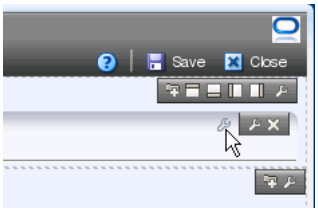
1. If it is not currently running, start the WebCenter Portal: Spaces application to which you added a Content Presenter display template. This application must have a Content Server connection that Content Presenter can use. If you just completed the setup procedures described in the previous sections, click the **Back to the Space** link in the upper right corner of the Administration page.
2. Navigate to the space in which you want to use the new Content Presenter display template. To do this, select the Spaces tab, and then select a space that you own or can edit.
3. Optionally, create a new page to test the display template. Select **Create a New Page** from the Page Actions menu. The, select **Save** and **Close**.
4. In the new or existing page, select **Edit Page** from the Page Actions menu.
5. Select an **Add Content** button in the part of the page where you want to add a Content Presenter.
6. In the Add Content dialog, open the **Content Management** folder, as shown in Figure 29–9.

Figure 29–9 Opening the Document Management Folder



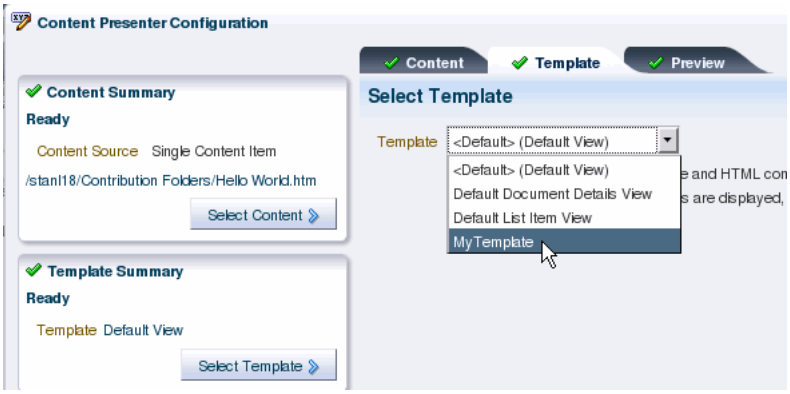
7. In the Add Content dialog, click the **Add** button next to Content Presenter.
8. Close the Add Content dialog.
9. Select the task flow **Edit** icon in the Content Presenter tool bar (see Figure 29–10).

Figure 29–10 Selecting the Edit Icon



10. In the Content Presenter Configuration window, select the Template tab.
11. From the Template menu, select your Content Presenter display template, as shown in Figure 29–11.

Figure 29–11 Selecting the Content Presenter Display Template



The Content Presenter template will be applied to the Content Presenter instance. For more information on using Content Presenter and display templates, see Section 31.2, "Understanding the Documents Service Task Flows" and Section 29.3, "Using the Out-of-the-Box Display Templates".

29.6 What Happens at Runtime?

This section includes these topics:

- Section 29.6.1, "Runtime Overview"
- Section 29.6.2, "Identifying Display Templates for Selected Content Items"

29.6.1 Runtime Overview

After you have defined a Content Presenter display template, you can add a Content Presenter task flow to a page in your application, specifying the content and display template in the Content Presenter task flow parameters (see Section 31.6, "Modifying Content Task Flow Parameters" and Section 29.3, "Using the Out-of-the-Box Display Templates").

If you integrate the Documents Service in your application (see Chapter 30, "Integrating the Documents Service"), end users of your application can select content and your Content Presenter display templates in the Content Presenter Configuration dialog to include content on editable pages of the application, as described in "Publishing Content Using Content Presenter" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. To understand how Content Presenter identifies which display templates to expose in the Content Presenter Configuration dialog, see Section 29.6.2, "Identifying Display Templates for Selected Content Items."

29.6.2 Identifying Display Templates for Selected Content Items

At runtime, after a user selects content in the Content Presenter Configuration dialog, Content Presenter checks the Resource Manager to identify the display templates that are suitable for the selected content item(s), then exposes the list of valid templates on the Template page in the Content Presenter Configuration dialog for the user to select. See also Chapter 17, "Adding and Using the Resource Manager."

For *single content item* templates, Content Presenter generates a list of valid templates using the following precedence order:

1. Check the Resource Manager for template definitions matching the content item's content repository and content type.
2. Add template definitions matching the default content repository (`*`) and specified content type.
3. Add template definitions matching inherited content types.
4. Add template definitions matching the default content repository (`*`) and default content type (`*`).

For *multiple content item* templates, Content Presenter generates a list of valid templates using the following precedence order:

1. Check the Resource Manager for template definitions associated with the selected template category.
2. Add template definitions associated with the default category (`*`).

For example, if the user selects multiple content items (such as the children of a folder, or the results of a search), the Content Presenter Configuration dialog shows a list of categories and the list of templates associated with that category and the default category, based on the repository of the content items.

Note: When a user selects a template in the Content Presenter Configuration dialog, that template is used to display the selected content item(s) in the running application. If that template is later deleted from the Resource Manager for any reason, Content Presenter will automatically select a "closest match" template to display the content item(s) using the precedence order above.

29.7 Performance Considerations for Content Presenter Display Templates

When content nodes are retrieved for display in a Content Presenter display template, most content item node property values are retrieved immediately along with the node, but some are loaded later only if needed. Other than the performance difference, the selective loading of node property values is transparent to the user or developer.

As a Content Presenter display template developer, you can optimize performance of your template if you are aware when different property values are loaded. For example, a typical list display template will render faster if you refer only to properties that are immediately loaded when the node is first retrieved and avoid properties that are loaded later when needed.

A secondary consideration is dependent on how the node is retrieved: through search versus fetched by parent ID. A property that may be loaded immediately on node retrieval for searches (such as "Results of a Query") may be loaded later for other retrieval methods (such as "Contents Under a Folder"). Table 29–8 shows whether or not node properties are loaded immediately upon node retrieval, by retrieval mechanism.

For information about the node properties listed in Table 29–8, see the *Oracle WebCenter Content Idoc Script Reference Guide*.

Table 29–8 Loading of Node Properties By Node Retrieval Mechanism

	Loaded When Node Is First Retrieved?		
OCS Global Profile Properties	**GET BY PARENT ID ("Contents Under a Folder")**	**SEARCH ("Results of a Query")**	**GET BY UUID ("Single Content Item" and "List of Items")**
`VaultFileSize`	**N**	Y	Y
`dCheckoutUser`	Y	**N**	Y
`dCreateDate`	Y	Y	Y
`dDocAccount`	Y	Y	Y
`dDocAuthor`	Y	Y	Y
`dDocName`	Y	Y	Y
`dDocTitle`	Y	Y	Y
`dDocType`	Y	Y	Y
`dFormat`	Y	Y	Y

Table 29–8 (Cont.) Loading of Node Properties By Node Retrieval Mechanism

OCS Global Profile Properties	Loaded When Node Is First Retrieved?		
	GET BY PARENT ID ("Contents Under a Folder")	SEARCH ("Results of a Query")	GET BY UUID ("Single Content Item" and "List of Items")
dID	Y	Y	Y
dInDate	Y	Y	Y
dIsCheckedOut	Y	N	Y
dOutDate	Y	Y	Y
dReleaseDate	Y	N	Y
dReleaseState	Y	N	Y
dRevClassID	Y	N	Y
dRevLabel	Y	Y	Y
dRevRank	Y	N	Y
dRevisionID	Y	Y	Y
dSecurityGroup	Y	Y	Y
dStatus	Y	N	Y
dWebExtension	Y	Y	Y
dWorkflowState	N	N	Y
idcPrimaryFile	Y	Y	Y
idcRenditions	N	N	Y
xCollectionID	Y	Y	Y
xComments	Y	Y	Y
xForceFolderSecurity	Y	Y	Y
xHidden	Y	Y	Y
xInihibitUpdate	Y	Y	Y
xReadOnly	Y	Y	Y

30

Integrating the Documents Service

There are several ways to integrate content into a WebCenter Portal: Framework application. For more information, see Chapter 26, "Introduction to Integrating and Publishing Content."

This chapter describes how to integrate the Documents service into a Framework application to both add content to the application, and to provide end users with content and Documents service task flows built into the application to manage, display, and search documents at runtime.

This chapter includes the following sections:

- Section 30.1, "Introduction to the Documents Service"
- Section 30.2, "Setting Up Connections"
- Section 30.3, "Setting Security for the Documents Service"
- Section 30.4, "Using the Documents Service with Other WebCenter Services"
- Section 30.5, "Setting Parameters to Upload Files to Content Repositories"
- Section 30.6, "Using Adapters with the Documents Service"

30.1 Introduction to the Documents Service

The Documents service provides features for accessing, adding, and managing folders and files; configuring and viewing file and folder properties; and searching file and folder content in Content Server, Oracle Portal, or SharePoint content repositories. It provides these features through Documents service task flows, document components (links, inline frames, and images), wikis, and blogs.

> **Note:** The availability of SharePoint as a content repository requires the installation of the SharePoint adapter, as described in Section 3.6, "Installing the Oracle WebCenter Adapter for SharePoint." Administration for SharePoint is performed using WLST commands, not Oracle Enterprise Manager Fusion Middleware Control Console.

To integrate the Documents service into a Framework application, your WebCenter Portal administrator must set up a connection to the content repository that contains the documents you want to manage. If the content repository you wish to use requires authentication, ensure that you set up an external application when you configure the connection to your content repository, as discussed in Section 30.2, "Setting Up Connections."

30.2 Setting Up Connections

Before you can use the Documents service and task flows, you must first set up the connection to the Content Server, Oracle Portal, or SharePoint content repository that contains the documents you want to manage. You can reuse connections you've created or create new ones.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, refer to the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The choices made for application security and content repository connection authentication need to be compatible. For example, if the connection is configured to use External Application with only Public credentials defined, then there is no requirement for the application to enforce user authentication. For detailed information about connection configuration, see Section 27.2, "Configuring Content Repository Connections."

30.3 Setting Security for the Documents Service

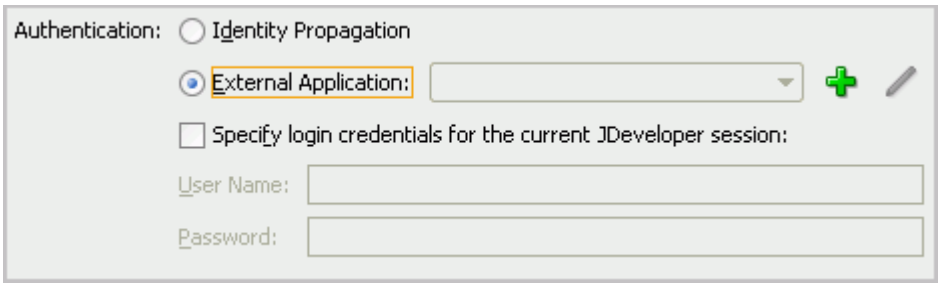
The Documents service task flows support the same security options that are supported by the content repository connections. This service can also use an external application with dedicated user accounts to access remote content repositories, such as an Oracle Content Server or Oracle Portal. For more information about using security with content repositories, see Section 28.4, "Securing a Content Repository Data Control." For information about using external applications, see Section 69.13, "Working with External Applications."

If you are using a content repository that handles its own authentication, such as Oracle Portal or Oracle Content Server, you can associate that content repository with an external application definition to allow for credential provisioning. You can modify your connection to your content repository to use an external application without shared or public credentials. When you do this, the External Application - Change Password task flow is automatically integrated to allow your end users to provide their credentials.

To register an external application for an existing content repository connection:

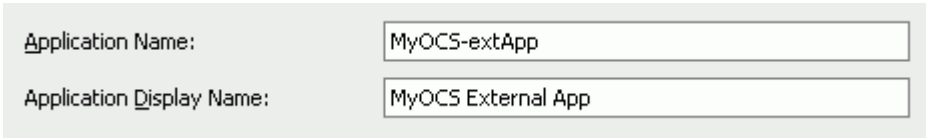
1. In the Application Resources pane, right-click your connection name and choose **Properties**.
2. In the Edit Content Repository Connection dialog box, under **Authentication**, select **External Application** (Figure 30–1).

Figure 30–1 Selecting External Application in the Content Repository Connection Dialog



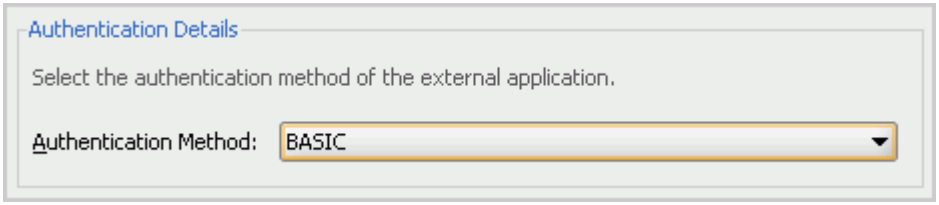
3. Click the green + (plus) sign to start the Register External Application wizard.
4. On the Name page, enter an **Application Name** and **Application Display Name**, (Figure 30–2) then click **Next**.

Figure 30–2 Naming the External Application



5. On the General page, under **Authentication Details**, choose **BASIC** from the **Authentication Method** list (Figure 30–3), then click **Next**.

Figure 30–3 Choosing the BASIC Authentication Method



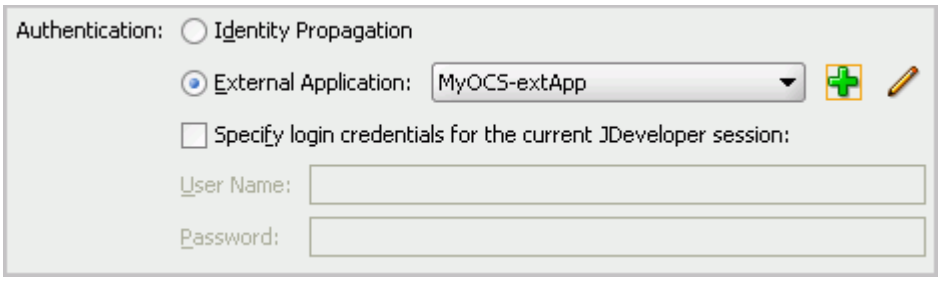
6. On the Additional Fields page, click **Next**.
7. On the Shared Credentials page, click **Next**.
8. On the Public Credentials page, select **Specify Public Credentials**, then enter the **Username** and **Password** for the content repository (Figure 30–4).

Figure 30–4 Specifying the Public Credentials



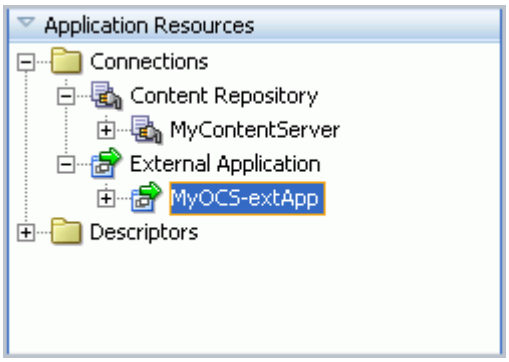
9. Click **Finish**. You'll notice that your connection now uses the external application for authentication (Figure 30–5).

Figure 30–5 Using the New External Application



10. In the Edit Content Repository Connection dialog box, click **OK**. In the Application Resources panel, you'll notice the external application now displays (Figure 30–6).

Figure 30–6 External Application in the Application Resources Panel



If you do not apply security, and the content repository requires a login to access the content, the user will not be able to authenticate, and thus will see only public content at runtime.

30.4 Using the Documents Service with Other WebCenter Services

You can use the Documents service with a variety of other WebCenter services. For example, you can add tags to documents in your content repository, search across your application and retrieve documents in the results, or track recent changes to the content repository.

You can see an example of how to use the Documents service with the Tags service in Section 44.3.3, "Optional Way to Show Tags on Pages."

To learn more about how you can use these services together, refer to Section 2.4, "Introducing Oracle WebCenter Portal: Services."

Note: When you integrate the Documents service with the Search service, the Search service returns results from *all* content repository connections.

30.5 Setting Parameters to Upload Files to Content Repositories

The Document Manager, Document Explorer, and Folder Viewer task flows allow you to upload files into content repositories. WebCenter Portal: Framework uses Apache MyFaces Trinidad to handle file upload from a browser to the application server.

To change the default settings of Apache MyFaces Trinidad, you can add three parameters to the `web.xml` file. To edit this file, open the **ViewController** project of your application. Under **Web Content**, open the `web.xml` file. In the Overview, navigate to **Application > Context Initialization Parameters**, then click the green plus sign (**+**) to add the parameters and their values (as described in Table 30–1) or simply update the code in the Source view. After you've made your changes, save the `web.xml` file, then restart Oracle JDeveloper.

Table 30–1 Apache MyFaces Trinidad Parameters

Parameter	Description
`org.apache.myfaces.trinidad.UPLOAD_MAX_MEMORY`	The maximum amount of memory in bytes that a single file can use when uploaded.
`org.apache.myfaces.trinidad.UPLOAD_MAX_DISK_SPACE`	The maximum amount of disk space in bytes that a single file can use when uploaded.
`org.apache.myfaces.trinidad.UPLOAD_TEMP_DIR`	The directory in which the file being uploaded is temporarily stored.

For more information, see the Apache MyFaces Trinidad documentation at `http://myfaces.apache.org/trinidad/devguide/fileUpload.html`.

30.6 Using Adapters with the Documents Service

To learn more about using existing adapters for the content repository connections, see Chapter 27, "Managing Content Repository Connections."

31

Adding Content Task Flows and Document Components to a Portal Page

There are several ways to integrate content into a WebCenter Portal: Framework Application. For more information, see Chapter 26, "Introduction to Integrating and Publishing Content."

In a Framework application, you can add content from one or more connected content repositories to the application, using Content Presenter or the Documents service task flows to display the content in a variety of ways. As you develop your application, you can view, edit, and manage that content using the content task flows, in the same way that end users of the application will do. For information about viewing, editing, and managing content in content task flows, see "Working with Content" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This chapter describes how to add content to a page in a Framework application using the content task flows and document components in the following sections:

- Section 31.1, "Understanding the Content Presenter Task Flow"
- Section 31.2, "Understanding the Documents Service Task Flows"
- Section 31.3, "Understanding Document Components"
- Section 31.4, "Adding a Selected Folder or File to a Page"
- Section 31.5, "Adding a Content Task Flow to a Page"
- Section 31.6, "Modifying Content Task Flow Parameters"
- Section 31.7, "Content Task Flow Parameters"

31.1 Understanding the Content Presenter Task Flow

Content Presenter enables you to precisely customize the selection and presentation of content in a WebCenter Portal application.

The Content Presenter task flow is available only when the connected content repository is Content Server and your WebCenter Portal administrator has completed the prerequisite configuration.

See Also:

- "Content Server Prerequisites" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- Section 27.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter"

The Content Presenter task flow allows you to select a single item of content, contents under a folder, a list of items, query for content, or select content based on the results of a Personalization Conductor scenario, and then select a template to render the content on a page in a WebCenter Portal application.

Content Presenter has no dependency on the Documents service for adding or managing the content it displays.

To add the Content Presenter task flow to a page, see Section 31.4, "Adding a Selected Folder or File to a Page" and Section 31.5, "Adding a Content Task Flow to a Page."

Note: If a page is based on a page template that includes a Content Presenter task flow to display a content item, the page template developer must manually modify the task flow properties to display the content in the page template, as follows:

- `datasource`: *`connection_name`*`#dDocName:`*`content_ID`*
- `datasourceType`: `dsTypeSingleNode`
- `taskFlowInstId`: A unique identifier

For more information about Content Presenter task flow properties, see Table 31–4, " Content Presenter Task Flow Parameters"

For information about creating a page template, see Section 12.2.2, "How to Create a Page Template."

31.2 Understanding the Documents Service Task Flows

The Documents service task flows provide a variety of formats to display folders and files, including wikis and blogs, on a page in a Framework application. You can choose the task flows appropriate for your application to provide features for accessing, adding, and managing folders and files; configuring and viewing file and folder properties; and searching file and folder content in Content Server, Oracle Portal, or SharePoint content repositories.

Note: The availability of SharePoint as a content repository requires the installation of the SharePoint adapter, as described in Section 3.6, "Installing the Oracle WebCenter Adapter for SharePoint." Administration for SharePoint is performed using WLST commands, not Oracle Enterprise Manager Fusion Middleware Control Console.

To add a Documents service task flow to a page in a Framework application, your WebCenter Portal administrator must first integrate the Documents service into the application by establishing a content repository connection, as described in Chapter 30, "Integrating the Documents Service" and Chapter 27, "Managing Content Repository Connections."

Using Documents service task flows and document components (such as links, previews, and images), you can add content to the application, and also provide end users with content and Documents service task flows built into the application to manage, display, and search documents at runtime.

Table 31–1 provides an overview of the Documents service task flows.

Table 31–1 Documents Service Task Flows

	Folder and File Listings	Individual Folders	Individual Files
Document Explorer task flow. Displays folders and files in two panes, combining the functionality of the Document Navigator and Folder Viewer task flows. It provides in-place previewing and editing, and robust management capabilities with an interface that should be familiar to users of Windows Explorer. Size: medium to full page width. See Figure 31–1.	X	X	
Document List Viewer task flow. Displays folders and files in a single pane as a flat listing. It provides preview and editing in separate window, and some management capabilities. Size: narrow to medium page width. See Figure 31–2.	X	X	
Document Manager task flow. Displays folders and files as specified by its `layout` property: `explorer`, `table`, or `treeTable`. The `explorer` layout is identical to the Document Explorer task flow, without the properties `showDocuments`, `showFolders`, and `treeNavCollapsed` . See Figure 31–3, Figure 31–4, and Figure 31–5.	X	X	
Document Navigator task flow. Displays a nested hierarchy of folders and files in a single pane, providing expand and collapse on folders to show the full hierarchy. Size: narrow to medium page width. See Figure 31–6.	X	X	
Folder Viewer task flow. Displays the contents of a folder in a single pane as a flat listing, providing in-place preview and editing, and robust document management capabilities with a straightforward interface that should be familiar to Windows users. Size: medium to full page width. See Figure 31–7.	X	X	
Recent Documents task flow. Displays a list of the files most recently created or modified in the current folder. See Figure 31–8.	X		
Document Viewer task flow. Displays a preview of a file, or file properties for files that do not support a preview. See Figure 31–9, Figure 31–10, Figure 31–11, and Figure 31–12.		X[1]	X
Document Mini Properties task flow. Displays the Basic properties of a file in a read-only view. See Figure 31–13.		X[1]	X
Document Properties task flow. Displays both Basic and Advanced properties of a file, along with an **Edit** button to allow you to modify property values. See Figure 31–14.		X[1]	X
Rich Text Editor task flow. Displays an HTML or wiki document in the Rich Text Editor.		X	X
Document Upload task flow. Displays the Upload dialog to allow users to upload new documents to the page.		X	X

Table 31–1 (Cont.) Documents Service Task Flows

	Folder and File Listings	Individual Folders	Individual Files
Document Version History task flow. Displays a list of versions of a file, allowing for deletion of a selected version. See Figure 31–15.			X
AutoVue task flow. Displays AutoVue markup for a file in a table of hyperlinked markup names (see "Collaborating on Documents Using AutoVue" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*).			X

[1] To show folder information in a Document Viewer, Document Mini Properties, or Document Properties task flow, you can set the task flow's `resourceID` property to the ID of the target folder. See Section 31.7, "Content Task Flow Parameters."

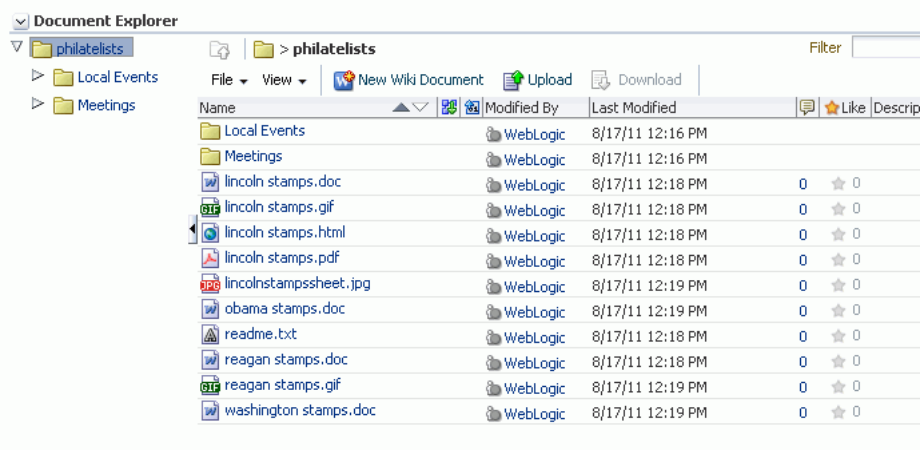
To add a Documents service task flow to a page, see Section 31.4, "Adding a Selected Folder or File to a Page" and Section 31.5, "Adding a Content Task Flow to a Page."

For detailed information about each Documents service task flow, click the links below to display the pertinent sections in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

- Understanding the Document Explorer Task Flow

  Figure 31–1 shows an example of the Document Explorer task flow. For information about setting properties for this task flow, see Section 31.7.2, "Document Explorer Task Flow Parameters."

Figure 31–1 Document Explorer Task Flow



- Understanding the Document List Viewer Task Flow

  Figure 31–2 shows an example of the Document List Viewer task flow. For information about setting properties for this task flow, see Section 31.7.3, "Document List Viewer Task Flow Parameters."

Figure 31–2 Document List Viewer Task Flow



- Understanding the Document Manager Task Flow

  Figure 31–3, Figure 31–4, and Figure 31–5 show examples of the three different layouts for the Document Manager task flow. The `explorer` layout (default) is identical to the Document Explorer task flow, without the properties `showDocuments`, `showFolders`, and `treeNavCollapsed` . For information about setting properties for this task flow, see Section 31.7.4, "Document Manager Task Flow Parameters."

Figure 31–3 Document Manager Task Flow: explorer Layout

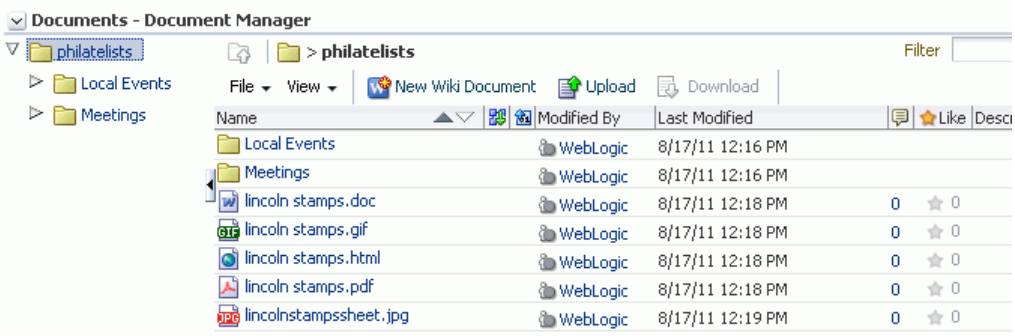


Figure 31–4 Document Manager Task Flow: table Layout

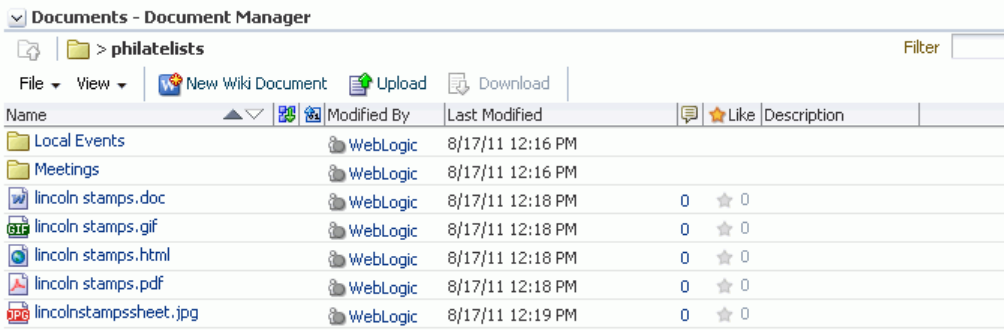
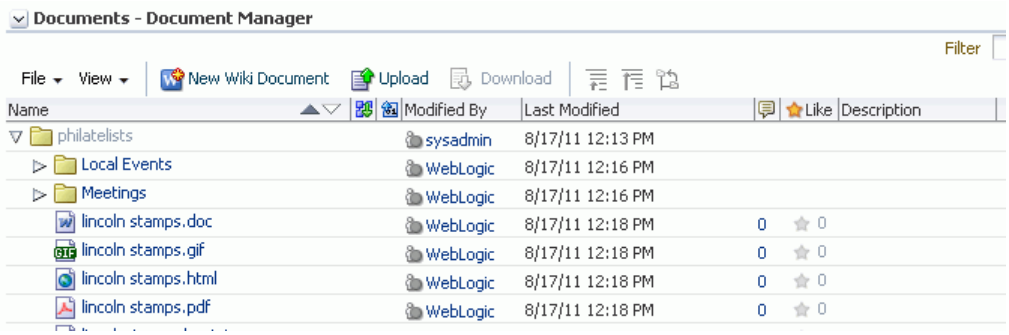


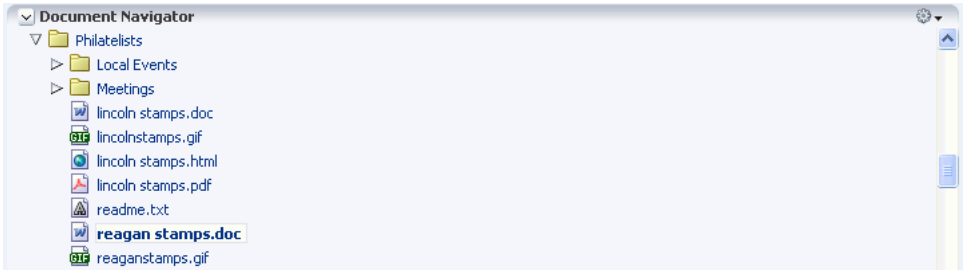
Figure 31–5 Document Manager Task Flow: treeTable Layout



- Understanding the Document Navigator Task Flow

  Figure 31–6 shows an example of the Document Navigator task flow. For information about setting properties for this task flow, see Section 31.7.5, "Document Navigator Task Flow Parameters."

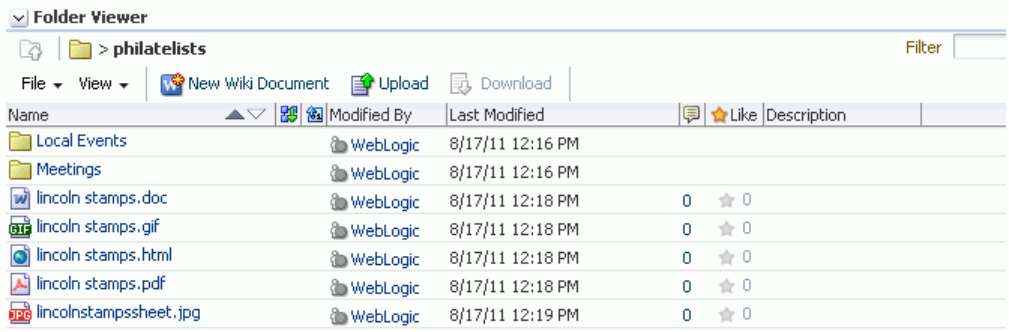
Figure 31–6 Document Navigator Task Flow



- Understanding the Folder Viewer Task Flow

  Figure 31–7 shows an example of the Folder Viewer task flow. For information about setting properties for this task flow, see Section 31.7.6, "Folder Viewer Task Flow Parameters."

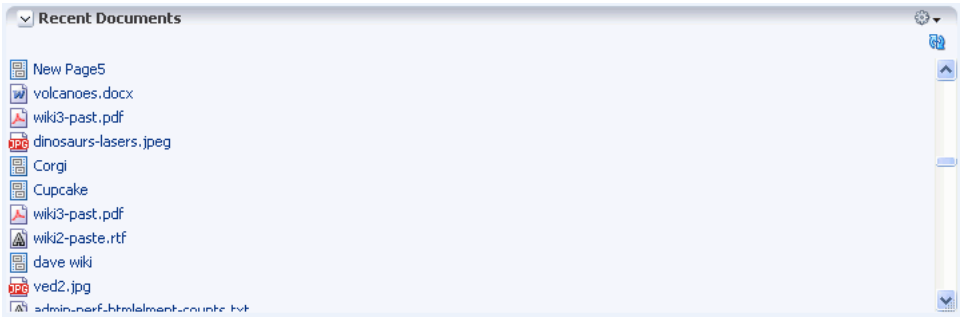
Figure 31–7 Folder Viewer Task Flow



- Understanding the Recent Documents Task Flow

  Figure 31–7 shows an example of the Recent Documents task flow. For information about setting properties for this task flow, see Section 31.7.7, "Recent Documents Task Flow Parameters."

Figure 31–8 Recent Documents Task Flow



- Understanding the Document Viewer Task Flow

  Figure 31–9, Figure 31–10, Figure 31–11, and Figure 31–12 show examples of the Document Viewer task flow for different file types. For information about setting properties for this task flow, see Section 31.7.8, "Document Viewer Task Flow Parameters."

Figure 31–9 Document Viewer Task Flow for Microsoft Office Word Document

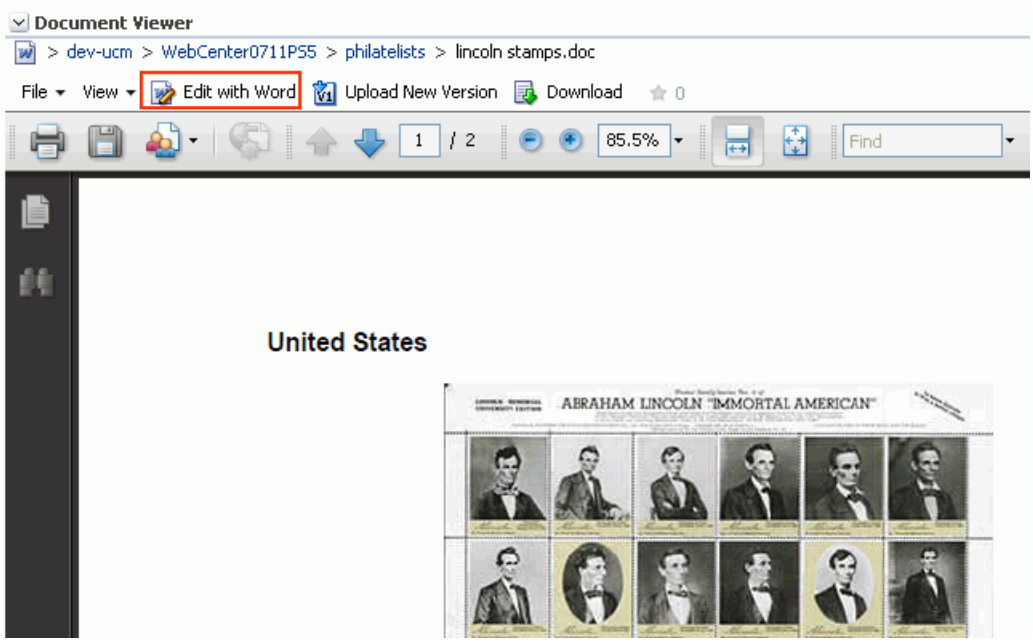


Figure 31–10 Document Viewer Task Flow for Image File



Figure 31–11 Document Viewer Task Flow for Wiki Document

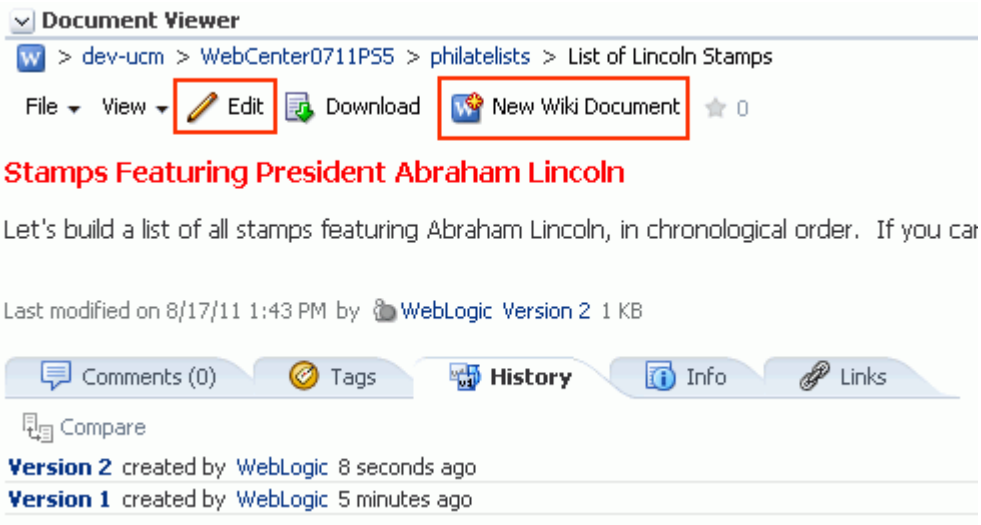
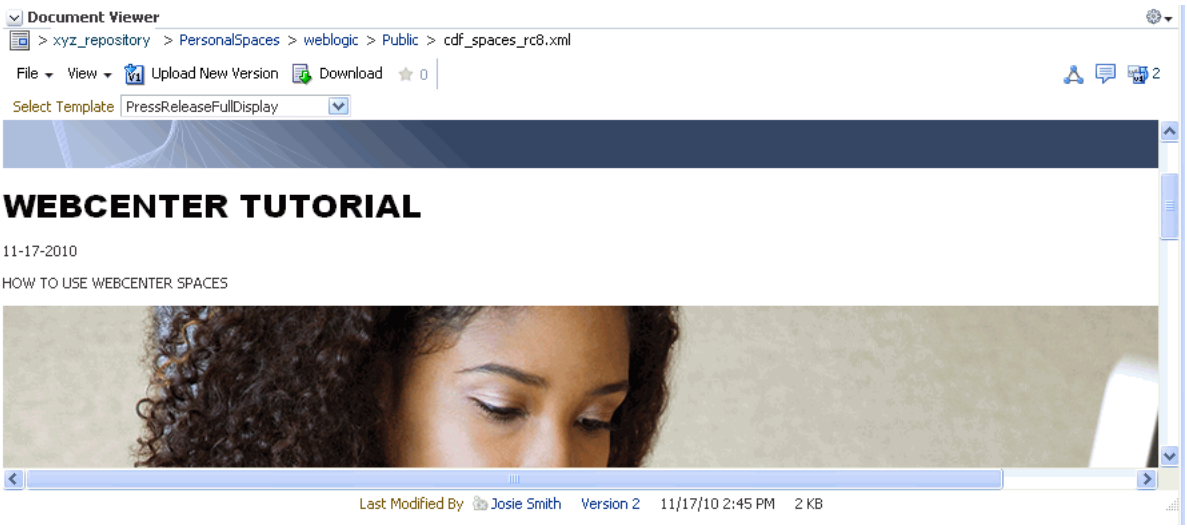


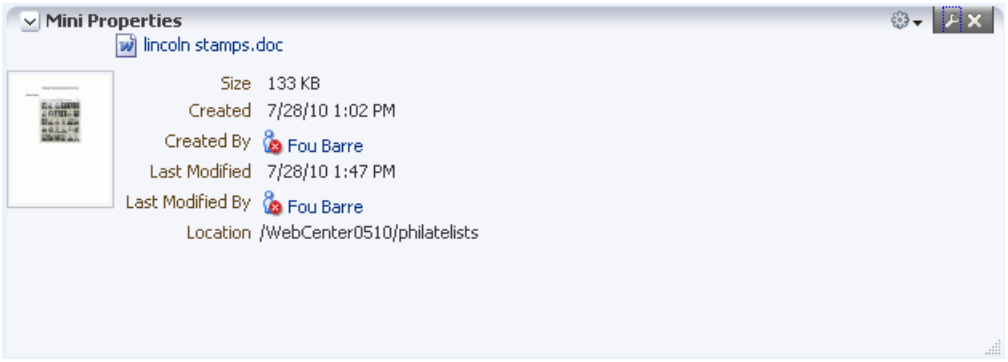
Figure 31–12 Document Viewer Task Flow for Site Studio File



- Understanding the Document Mini Properties Task Flow

  Figure 31–13 shows an example of the Document Mini Properties task flow. For information about setting properties for this task flow, see Section 31.7.9, "Document Mini Properties Task Flow Parameters."

Figure 31–13 Mini Properties Task Flow for File



- Understanding the Document Properties Task Flow

  Figure 31–14 shows an example of the Document Properties task flow. For information about setting properties for this task flow, see Section 31.7.10, "Document Properties Task Flow Parameters."

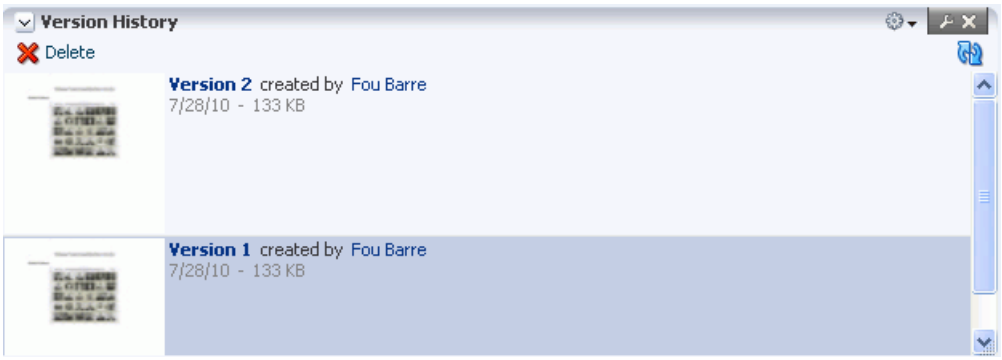
Figure 31–14 Document Properties Task Flow for File



- Understanding the Document Version History Task Flow

  Figure 31–15 shows an example of the Document Version History task flow. For information about setting properties for this task flow, see Section 31.7.13, "Document Version History Task Flow Parameters."

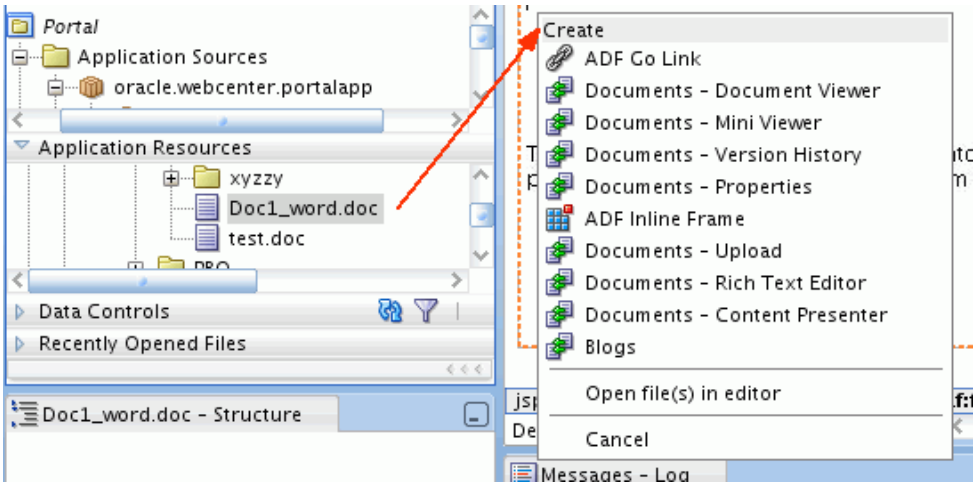
Figure 31–15 Version History Task Flow for File



31.3 Understanding Document Components

The Documents service provides features for adding document components to a portal page. As described in Section 31.4, "Adding a Selected Folder or File to a Page", you can add a document component to a page as a container for a selected folder or file (Figure 31–16).

Figure 31–16 Adding File to Portal Page



Document components enable you to display an individual file on a page in a variety of ways, depending on the file type:

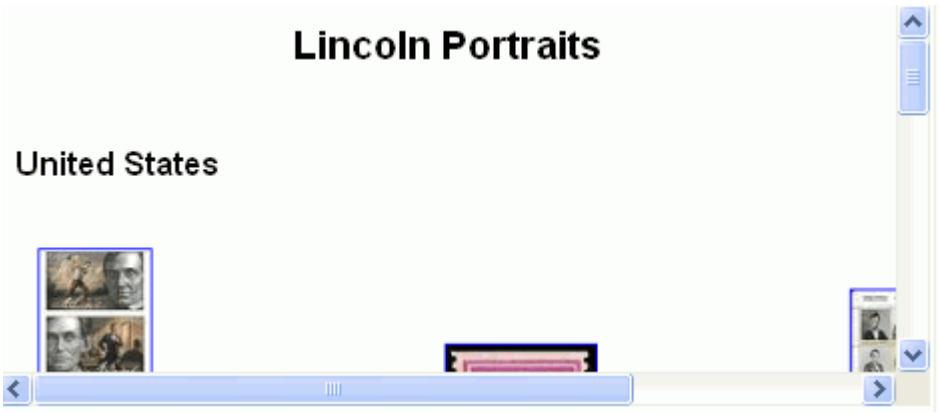
- A **Link** displays the name of a selected file as a link, which end users can click to display the file content in its native application (Figure 31–17).

Figure 31–17 Link Component (for Microsoft Word file)



- An **Inline Frame** displays the content of a selected file as a preview on the page (Figure 31–18).

Figure 31–18 Inline Frame Component (for HTML file)



- An **Image** displays a selected file as an image on the page (Figure 31–19).

Figure 31–19 Image Component (for JPEG file)



To add a document component to a page in a Framework application, you must first integrate the Documents service into the application by establishing a connection to a content repository, as described in Chapter 30, "Integrating the Documents Service."

Table 31–2 shows the document components available for the different content types.

Table 31–2 Components for Folders and FIles

Content Type	Document Components
Folder	Link (ADF Go Link)
Documents of various types (XML, PDF, JAVA, TXT, DOC, XLS, HTM)	Link (ADF Go Link)
Documents that can be rendered in a browser (HTML, flash, PDF and image)	Inline frame (ADF Inline Frame)
Images (PNG, JPG, GIF)	Link (ADF Go Link) Inline frame (ADF Inline Frame) Image (ADF Image)

To display a folder or file using one of these document components, see Section 31.4, "Adding a Selected Folder or File to a Page."

31.4 Adding a Selected Folder or File to a Page

If you want to add a content task flow to a page, independent of a specific folder or file, refer to Section 31.5, "Adding a Content Task Flow to a Page." This section describes how to display an individual folder or file from a connected content repository on a page, as shown inFigure 31–20 and Figure 31–21, by choosing the required container for the selected folder or file.

Figure 31–20 Adding Folder to Portal Page

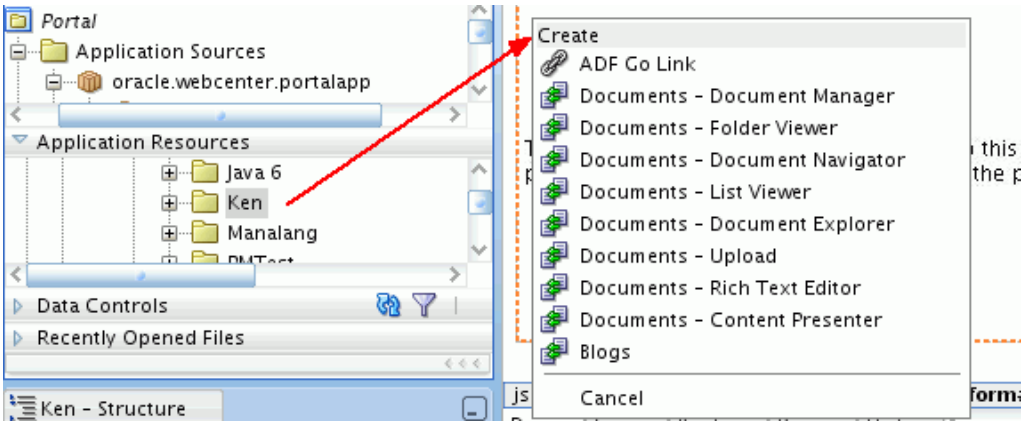


Figure 31–21 Adding File to Portal Page

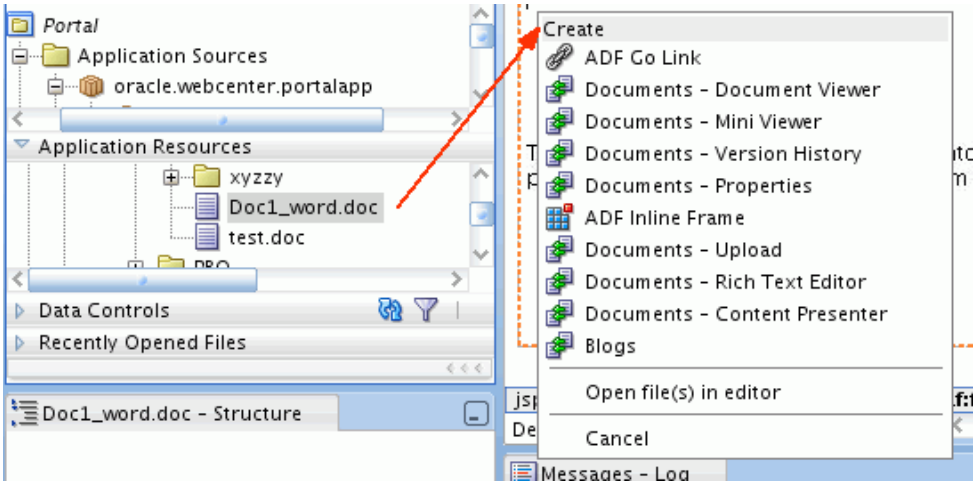


Table 31–3 lists all containers available for displaying folders and files on a portal page:

Table 31–3 Containers for Adding Individual Folders and Files to a Page

Root Folder	Folders	Individual Files
▪ Document Explorer ▪ Document List Viewer ▪ Document Manager ▪ Document Navigator ▪ Folder Viewer ▪ Recent Documents ▪ Document Viewer[1] ▪ Document Mini Properties[1] ▪ Document Properties[1] ▪ Document Upload ▪ Rich Text Editor	▪ Content Presenter ▪ Document Explorer ▪ Document List Viewer ▪ Document Manager ▪ Document Navigator ▪ Folder Viewer ▪ Document Viewer[1] ▪ Document Mini Properties[1] ▪ Document Properties[1] ▪ Document Upload ▪ Rich Text Editor	▪ Content Presenter ▪ Document Viewer ▪ Document Mini Properties ▪ Document Properties ▪ Rich Text Editor ▪ Document Upload ▪ Document Version History ▪ AutoVue ▪ Link component ▪ Inline Frame component ▪ Image component

[1] To show folder information in a Document Viewer, Document Mini Properties, or Document Properties task flow, you can set the task flow's `resourceID` parameter to the ID of the target folder. See Section 31.7, "Content Task Flow Parameters."

For more information about any of the containers listed in Table 31–3, see Section 31.1, "Understanding the Content Presenter Task Flow", Section 31.2, "Understanding the Documents Service Task Flows", and Section 31.3, "Understanding Document Components."

Note: When an item that is stored in a content repository is dropped onto a page, the Framework application instructs the browser to check whether the content is up-to-date before a cached copy, if available, displays in the browser. The default content validation process supports collaboration use cases that require real-time content exchange.

For static content or content that does not change frequently, content validation checks incur an unnecessary performance overhead. For static content, the following is recommended:

- **Oracle Content Server**: Use Content Presenter to display static content stored in Content Server. Content Presenter shows only the latest released version of content; it does not show any unreleased content that is in workflow
- **Other content repositories**, such as SharePoint: Ask your administrator to override the default content cache setting by editing the Web Tier `mod_wl_ohs` configuration file (`mod_wl_ohs.conf`). This is fully explained in "Setting the Cache-Control Response Header" in the white paper "Integrating the SharePoint 2007 Adapter with WebCenter Spaces" available from Oracle Technology Network at `http://www.oracle.com/technetwork/middleware/webcenter/overview/index.html`.

To add a selected folder or file to a page in a Framework application (see Section 5.2.1, "How to Create a Framework Application"):

1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security.
2. If you want to add a Documents service task flow, create a content repository connection, which adds the Documents service to your application (see Chapter 30, "Integrating the Documents Service").
3. Create or open a page in your application where you want to add the task flow (see Section 5.3, "Adding Pages to a Portal").
4. Consider whether or not you want to allow end users of your application to edit the task flow that displays the folder or file:
   - To enable runtime editing of the page, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."
   - To allow end users of your application to customize a Content Presenter or Document List Viewer task flow using the **Edit** action, add the task flow inside a Show Detail Frame Component (see Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components").

5. In the **Application Resources** panel, navigate to **Connections**, then **Content Repository**.
6. Locate the name of your connection, then expand it if desired to navigate to a specific folder or file.
7. Drag the folder or file to the page, and release the mouse button to display a menu of available task flows and document components, as shown in Figure 31–22 (root folder), Figure 31–23 (subfolder), and Figure 31–24 (individual file).

Figure 31–22 Adding Content Repository Root Folder to Portal Page

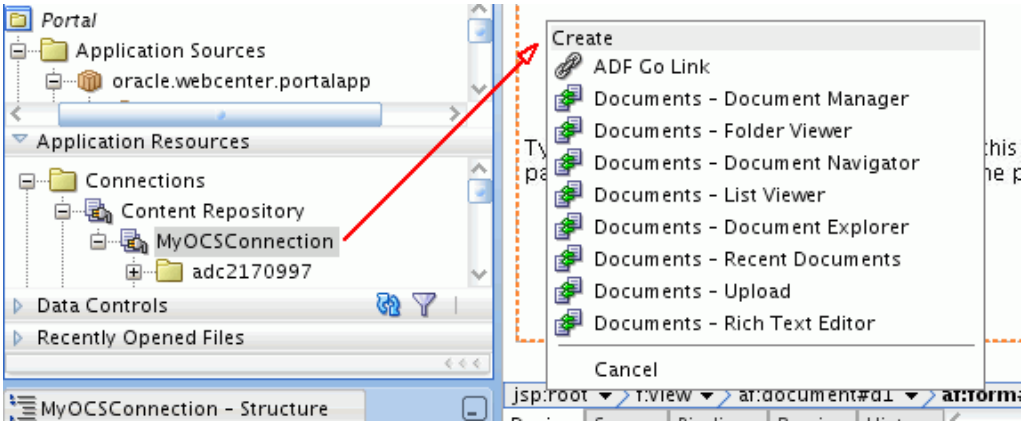


Figure 31–23 Adding Subfolder to Portal Page

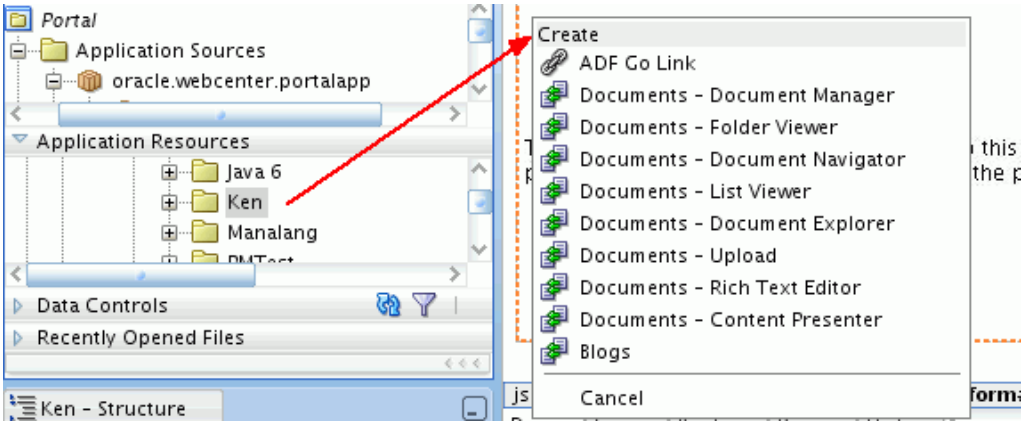
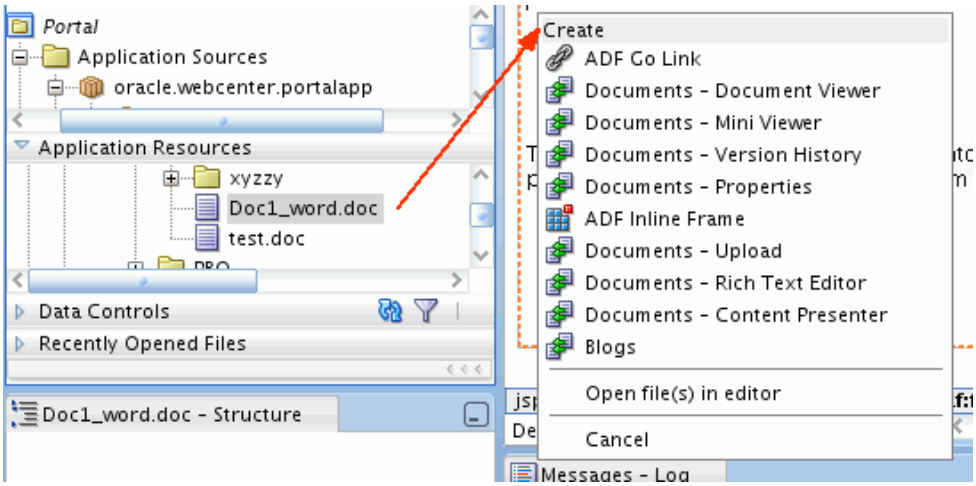


Figure 31–24 Adding File to Portal Page



8. Choose the content task flow or document component that you wish to use. If the folder or file you are adding to your page does not support a particular task flow, the task flow does not display in this menu.

> **Note:** The Content Presenter task flow is available only when the connected content repository is Content Server. When you select the Content Presenter task flow, the folder or file displays in a default display template for its type.

9. In the Edit Task Flow Binding dialog that displays when you select a task flow, click **OK** to accept the default task flow parameters, or modify the parameters as desired. For more information, see Section 31.7, "Content Task Flow Parameters."

10. Save your page and the page definition file, then run your page to your browser by right-clicking the page (not the page definition file) and choosing **Run**.

For information about viewing, editing, and managing content in content task flows, see "Working with Content" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

31.5 Adding a Content Task Flow to a Page

You may want to add an "empty" content task flow to a page, containing no folders or files. For example, in cases where you do not have a content repository connection, or want to use an EL expression as a parameter to dynamically bind to a target item.

See Chapter 26, "Introduction to Integrating and Publishing Content" for information about each content task flow to help you select the task flow most appropriate for your needs.

To add a content task flow to a page in a Framework application:

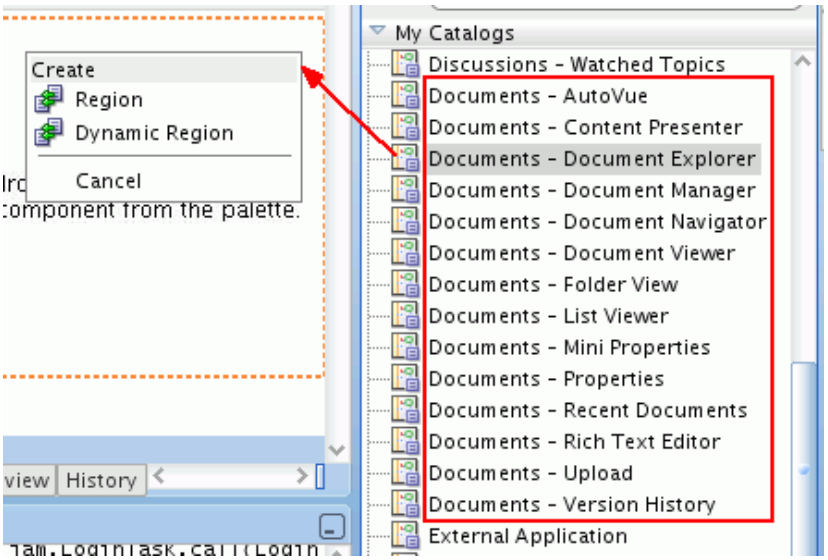
1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security.

2. If you want to add a Documents service task flow, see Chapter 30, "Integrating the Documents Service".

3. Create or open a page in your application where you want to add the task flow (see Section 5.3, "Adding Pages to a Portal").

4. Before you add a task flow to your page, consider whether or not you want to allow end users of your application to edit the task flow:
   - To enable runtime editing of the page, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."
   - To allow end users of your application to customize a Content Presenter or Document List Viewer task flow using the **Edit** action, add the task flow inside a Show Detail Frame Component (see Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components").
5. In the **Resource Palette**, expand **My Catalogs**, then **WebCenter Portal - Services Catalog**. Under **Task Flows**, select the desired content task flow and drag it onto your page (Figure 31–25), and release the mouse button.

Figure 31–25 Adding Standalone Documents Service Task Flow to JSF Page

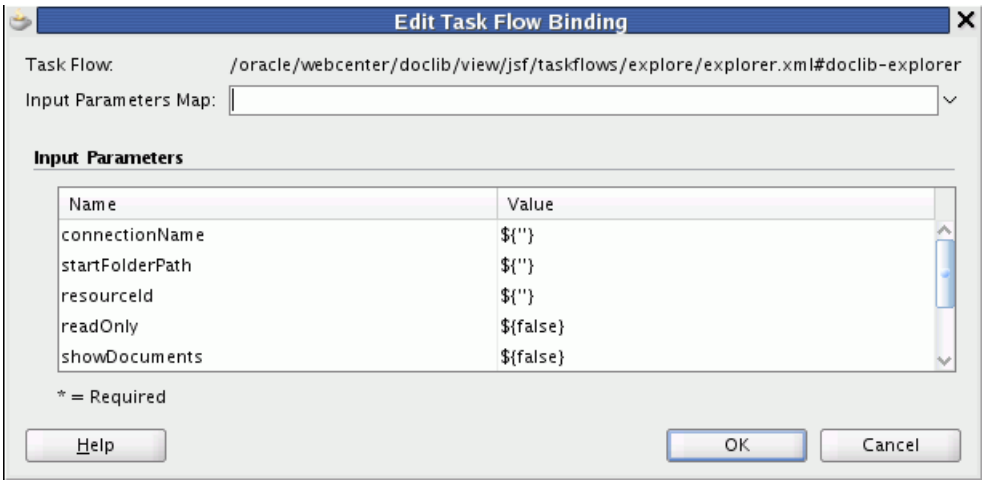


Note: To allow end users of your application to customize a Content Presenter or Document List Viewer task flow using the **Edit** action, drag the task flow inside a Show Detail Frame Component (see Section 20.1.6, "How to Enable Component Customization Using Show Detail Frame Components").

Note: If you display a wiki page in a Framework application using a Content Presenter display template, by default any links within that wiki page are displayed in the Document Viewer. If you want to display wiki page links using Content Presenter, you must edit the `adf-config.xml` file. For more information, see Section 32.3.5, "Displaying Wiki Page Links Within Content Presenter."

6. From the menu that displays, choose **Region** to open the Edit Task Flow Binding dialog (Figure 31–26).

Figure 31–26 Specifying Parameters for Document Explorer Task Flow



7. In the Edit Task Flow Binding dialog, click **OK** to accept the default task flow parameters, or modify the parameters as desired. For more information, see Section 31.7, "Content Task Flow Parameters."
8. Save your page and the page definition file, then run your page to your browser by right-clicking the page (not the page definition file) and choosing **Run**.

After you add a content task flow to a page, you can subsequently modify the parameters if required. See Section 31.6, "Modifying Content Task Flow Parameters".

For information about viewing, editing, and managing content in content task flows, see "Working with Content" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Note: For information about adding custom actions to task flows, see Section 22.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow."

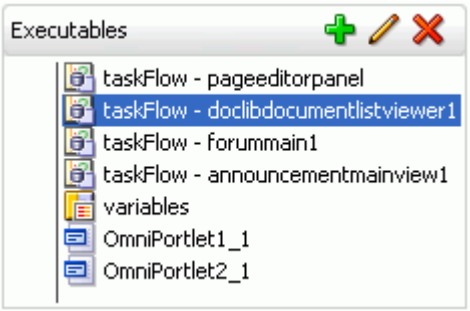
Note: By default, the task flow displays as full screen at runtime. You can use ADF layout components to modify this layout. To learn more about ADF layout components, refer to the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

31.6 Modifying Content Task Flow Parameters

To modify the parameters of a content task flow after you have created the region:

1. Go to the Bindings view of your page (click the **Bindings** tab next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Select this task flow. Figure 31–27 shows an example of a Document List Viewer task flow in the **Executables** section.

Figure 31–27 Document List View Task Flow in the Bindings View

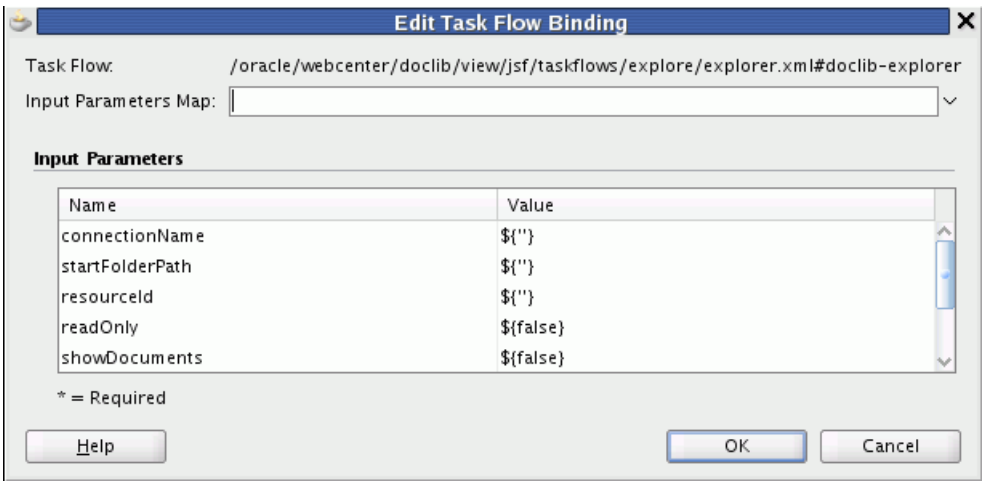


3. Next to the **Executables** heading, click the pencil icon to display the Edit Task Flow Binding dialog.
4. In the Edit Task Flow Binding dialog, specify parameter settings for the task flow. For more information, see Section 31.7, "Content Task Flow Parameters."

Note: For a Content Presenter task flow:

- If you want to allow end users of your application to select the content and display template using the Content Presenter Configuration dialog at runtime, you can leave all parameter values blank except for `taskFlowInstId`. Enter a task flow instance ID that is unique to your application (for example, `contentPresenterInstance1`).
- If you specify the content and a display template, note that Content Presenter does not support non-ASCII characters in files that are encoded using the non-UTF-8 character encoding. When users preview such files in Content Presenter, non-ASCII characters appear garbled.

Figure 31–28 Task Flow Binding Parameters



5. Click **OK** to add the parameters.

31.7 Content Task Flow Parameters

Each content task flow has its own set of parameters. You configure these values when you add the task flow to a page.

The following sections describe the parameters for the content task flows:

- Section 31.7.1, "Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates"
- Section 31.7.2, "Document Explorer Task Flow Parameters"
- Section 31.7.3, "Document List Viewer Task Flow Parameters"
- Section 31.7.4, "Document Manager Task Flow Parameters"
- Section 31.7.5, "Document Navigator Task Flow Parameters"
- Section 31.7.6, "Folder Viewer Task Flow Parameters"
- Section 31.7.7, "Recent Documents Task Flow Parameters"
- Section 31.7.8, "Document Viewer Task Flow Parameters"
- Section 31.7.9, "Document Mini Properties Task Flow Parameters"
- Section 31.7.10, "Document Properties Task Flow Parameters"
- Section 31.7.11, "Rich Text Editor Task Flow Parameters"
- Section 31.7.12, "Document Upload Task Flow Parameters"
- Section 31.7.13, "Document Version History Task Flow Parameters"

31.7.1 Content Presenter Task Flow Parameters and Out-of-the-Box Display Templates

The Content Presenter task flow is available when the connected content repository is Content Server. This task flow displays selected documents in out-of-the-box or custom display templates.

See Also:

- Section 31.1, "Understanding the Content Presenter Task Flow"
- Section 31.5, "Adding a Content Task Flow to a Page"

Table 31–4 describes the Content Presenter task flow parameters. Table 31–5 and Table 31–6 describe the out-of-the-box display templates for single and multiple content items, which can be specified as values for the `templateView` and `templateCategory` parameters.

Note: When you specify the content and a display template, note that Content Presenter does not support non-ASCII characters in files that are encoded using the non-UTF-8 character encoding. When users preview such files in Content Presenter, non-ASCII characters appear garbled.

If you want to allow end users of your application to select the content and display template using the Content Presenter Configuration dialog at runtime, you can leave all parameter values blank except for `taskFlowInstId`. Enter a task flow instance ID that is unique to your application (for example, `contentPresenterInstance1`).

At runtime, the parameters of a Content Presenter task flow on an editable page can be edited in two ways by end users with appropriate permissions:

- Click the configuration **Edit** icon for the task flow to display the Content Presenter Configuration dialog, where they can specify content and a display template. For end user information about the Content Presenter Configuration dialog, see the "Publishing Content Using Content Presenter" chapter in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- Click the properties **Edit** icon for the task flow to display the Component Properties dialog where they can set and modify task flow parameters. Parameters set in this way are generally intended for use at design time by developers creating a Framework application, or for advanced users who want to bind a parameter to an EL expression. If an end user modifies a parameter value in the Component Properties dialog, the new value overrides the value specified in the Content Presenter Configuration dialog.

Table 31–4 Content Presenter Task Flow Parameters

Parameter	Description

`datasource`	The data source of the content. The value depends on the value of `datasourceType`:
	■ When `datasourceType=dsTypeSingleNode`, set `datasource` to a single document node identifier in the format:
	`connection_name``#dDocName:`*`content_id`*
	Example:
	`myconnection.us.oracle.com#dDocName:STAN_IDC-007619`
	■ When `datasourceType=dsTypeFolderContents`, set `datasource` to a single folder node identifier in the format:
	`connection_name``#dCollectionID:`*`collection_id`*
	Example:
	`myconnection.us.oracle.com#dCollectionID:45535`
	■ When `datasourceType=dsTypeQueryExpression`, set `datasource` to a CMIS search expression.
	Examples:
	`select * from cmis:document where cmis:name like \'test%\'`
	`connectionName=`*`connection_name`*`#select * from cmis:document where cmis:createdBy = \'weblogic\'`
	Note: If `connectionName` is not specified, then the primary connection will be used.
	For more information about how to format the query and for more examples, see *Oracle Fusion Middleware Content Management REST Service Developer's Guide*.
	■ When `datasourceType=dsTypeMultiNode`, set `datasource` to a set of comma-delimited document node identifiers in the format:
	`connection_name``#dDocName:`*`content_id`*`, `*`connection_name`*`#dDocName:`*`content_id`*`, ...`
	Example: `myconn#dDocName:DOCUMENT_ID_12345,myconn#dDocName:DOCUMENT_ID_56789`
	■ When `datasourceType=dsTypeScenarioResults`, set `datasource` to point to a scenario in the format:
	`conductor-connection-name=`*`conductor_conn_name`*`,namespace=`*`scenario_namespace`*`,scenario-name=`*`scenario_name`*`,`*`inputparm1=value1,inputparm2=value2,`*` ...`
	Example:
	`conductor-connection-name=ConductorServiceLocal,namespace=CPNamespace,scenario-name=GetRelatedDocsScenario,topic=outdoors,interests=hiking`
	For information about the Personalization Conductor and scenarios, see Chapter 67, "Personalizing WebCenter Portal Applications."

Table 31–4 (Cont.) Content Presenter Task Flow Parameters

Parameter	Description
`datasource` (cont'd)	where: ■ *`connection_name`* = the name of the content repository connection. ■ *`content_id`* = the `Content ID` for the content specified on the content information page for the item in Content Server. ■ *`collection_id`* = the `dCollectionID` found in the URL for the folder information page in Content Server, or the `CollectionID` value on the content server. ■ *`conductor_connection_name`* = the name of the URL connection that points to the Personalization Conductor (this name must start with `Conductor`). This value must match the `Reference name` attribute value in the `connections.xml` file for this URL connection. For more information, see "Configuring Content Presenter" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* and Appendix A, "Files for WebCenter Portal: Framework Applications" (for details about `connections.xml`). ■ *`namespace`* = the name of the namespace that contains the specified scenario. ■ *`scenario_name`* = the name of the scenario that Content Presenter will be using.
`datasourceType`	The data source type of the content. Valid values are: ■ `dsTypeSingleNode`: A single content item. ■ `dsTypeFolderContents`: The contents of a folder. ■ `dsTypeQueryExpression`: The results of a query. ■ `dsTypeMultiNode`: An ordered list of content items. ■ `dsTypeScenarioResults`: The results of the specified scenario. See Chapter 67, "Personalizing WebCenter Portal Applications."
`maxResults`	The maximum number of results to display when `datasourceType` is `dsTypeQueryExpression`. Default: `100`
`regionTemplate`	Specifies whether the display template is a Site Studio region definition template. This value is valid only with Content Server 11*g* or higher: ■ Selected: Display template is a Site Studio region definition template. The `templateView` value is set to the Content ID of the region template. ■ Cleared (default): Display template is not a Site Studio region definition template. For information about creating and using Site Studio region templates, see "Publishing Content Using Content Presenter" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* and the Oracle Site Studio documentation library.

Table 31–4 (Cont.) Content Presenter Task Flow Parameters

Parameter	Description
`taskFlowInstId`	The unique identifier of this task flow instance, used internally to maintain the association of the task flow instance with its customization and personalization settings and to manage saved queries. The value is generated automatically when you add individual files or the content of folders to your page from the Application Resources Panel, displaying the content in a Content Presenter task flow. The value is blank when you add a Content Presenter task flow onto your page from the Resource Palette. Enter or edit this value to set the unique identifier for the task flow.
`templateCategory`	The display template category ID to use in rendering results for multiple content items. This ID may reference the default template category for an out-of-the-box display template (Table 31–6) or a custom category created for a display template for multiple content items that has been defined as described in Section 29.4, "Creating Content Presenter Display Templates".
`templateView`	The display template view ID to use in rendering results for single content items. Enter the view ID of a template that is configured in the Resource Manager for a specific content type, or for list-based templates by category ID. This ID may reference one of the out-of-the-box display templates (Table 31–5), a custom display template that has been defined as described in Section 29.4, "Creating Content Presenter Display Templates", or set to the contentID of a region template if the content is a region.

The out-of-the-box display templates provided by Oracle WebCenter are listed in Table 31–5 and Table 31–6.

Table 31–5 Out-of-the-Box Templates for Displaying Single Content Items

Single Content Item Display Templates	View ID	Description
Default Document Details View	`oracle.webcenter.content.templates.default.document.details`	Displays detailed information about any single content item including creation date, modification date, created by username, modified by username, path and any comments.
Default List Item View	`oracle.webcenter.content.templates.default.list.item`	Used by multiple content item views to display each individual item. Displays a single line with an icon and item name as a link that either displays or downloads the item when clicked.
Default View (default when no template is selected)	`oracle.webcenter.content.templates.default.detail`	Displays any single content item, either directly in the browser (images, HTML, text) or as a link that downloads the associated file when clicked.

Table 31–6 Out-of-the-Box Templates for Displaying Multiple Content Items

Multiple Content Item Display Templates	View ID	Description
Accordion View	`oracle.webcenter.content.templates.default.list.panel.accordion`	Displays multiple content items in an accordion format, where each item can be expanded to display its details.
Bulleted View	`oracle.webcenter.content.templates.default.list.bulleted`	Displays multiple content items in a bulleted list format. Only content items will be displayed; folder items will be omitted.
Bulleted with Folder Label View	`oracle.webcenter.content.templates.default.list.bulleted.label`	Displays multiple content items in a bulleted list format. The name of the folder containing the first item in the list will display as a label above the list. This template is intended to be used when datasourceType is set to dsTypeFolderContents to ensure that all items have the same parent folder. Only content items will be displayed; folder items will be omitted.
Carousel View	`oracle.webcenter.content.templates.default.list.carousel`	Displays multiple content items in a carousel format, where items can be browsed by moving a slider left or right.
Icon View	`oracle.webcenter.content.templates.default.list.tile`	Displays multiple content items in a tiled format with large icons and file names.
List View (default when no template is selected)	`oracle.webcenter.content.templates.default.list.simple`	Displays multiple content items in a simple list.
List with Details Panel View	`oracle.webcenter.content.templates.default.list.details.panel`	Displays multiple content items in a list on the left, with a panel to the right displaying the details of a selected item.
Sortable Table View	`oracle.webcenter.content.templates.default.list.tabular`	Displays multiple content items in a sortable table that includes the document name, date created, and date modified.
Tabbed View	`oracle.webcenter.content.templates.default.list.panel.tabbed`	Displays multiple content items as tabs that can be selected to display item details. content items in a simple list.

31.7.2 Document Explorer Task Flow Parameters

The Document Explorer task flow displays a list of folders and files in two panes: the left pane shows folders, and the right pane show the contents of the currently selected folder. It is a feature-rich documents task flow for viewing, managing, and collaborating on folders and files.

See Also:

- "Understanding the Document Explorer Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Explorer task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–7 describes the Document Explorer task flow parameters.

Table 31–7 Document Explorer Task Flow Parameters

Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`pageSize`	The maximum number of rows to show in the task flow. If the listing of folders and files in the task flow is larger than the specified number of rows, the task flow displays a scroll bar. Default: `27` Typical scenarios where you may wish to alter this value are: - The majority of end users have bigger screens, allowing for the display of more rows. - You want to fit this task flow into a smaller area of a page. **Note**: If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the task flow scroll bar with the application scroll bar.
`readOnly`	Specifies whether to disable and hide all content management operations: - `${true}`: Disable content management. - `${false}` (default): Expose content management to users.
`resourceId`	The currently focused resource. This value can be a folder ID or a document ID. The `resourceId` value is checked for coherence with the `connectionName` and `startFolderPath` values, and influences their default values when they are not explicitly specified. For example, if `resourceId` alone is specified, the `connectionName` value will be inherited from it and the startFolderPath will be either the resource itself for a folder or the parent folder for a file.

Table 31–7 (Cont.) Document Explorer Task Flow Parameters

Parameter	Description
`showDocuments`	Specifies whether the navigation tree shows documents and folders, or folders only: ■ `${true}`: Show documents and folders. ■ `${false}` (default): Show folders and hide documents.
`showFolders`	Specifies whether the navigation tree shows documents and folders, or documents only: ■ `${true}` (default): Show folders and documents. ■ `${false}`: Show documents and hide folders.
`startFolderPath`	The name of the folder to use as the root folder in the current task flow instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the task flow instance. There is no need to set this value for task flows that display the content of the current space's default root folder. But it is useful, for example, when you want the start folder to be other than a space's default root folder and when you want to display content from another space. Example: ■ `${'/PersonalSpaces/monty/Public'}` ■ `${'/WebCenterB5/Proj_X/Specs'}` You can specify an EL expression to set this value. Default: The root folder of the content repository configured with the specified connection for the current Space.
`treeNavCollapsed`	Specifies whether to collapse or expand the panel containing the tree navigation: ■ `${true}` (default): Collapse panel. ■ `${false}`: Expand panel.

31.7.3 Document List Viewer Task Flow Parameters

The Document List Viewer task flow displays folders and files in a single pane as a flat listing. In this task flow, users can navigate a folder hierarchy, and customize search queries. While this task flow may be useful for a specific need, its search functionality is replicated and enhanced by the Content Presenter task flow.

See Also:

- "Understanding the Document List Viewer Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document List Viewer task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–8 describes the Document List Viewer task flow parameters.

Table 31–8 Document List Viewer Task Flow Parameters

Task Flow Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.
`createdAfter`	A filtering value to limit the display of task flow content to folders and files created after a specified date and time. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZ'}`[1] Example: `${'2011-03-17T18:24:36.000+01:00'}`
`createdBefore`	A filtering value to limit the display of task flow content to folders and files created before a specified date and time. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZ'}`[1] Example: `${'2011-03-17T18:24:36.000+01:00'}`
`creator`	A filtering value to limit the display of task flow content to folders and files created by a particular user. Enter the user name as specified by the user login credentials. Only one user name may be entered. If no value is entered, then content created by any user is shown. Example: `${'monty'}`
`lastModifiedAfter`	A filtering value to limit the display of task flow content to folders and files last modified after a specified date and time. If no value is entered, then content modified in the last three months is shown. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`[1] Example: `${'2011-03-17T18:24:36.000+01.00'}`
`lastModifiedBefore`	A filtering value to limit the display of task flow content to folders and files modified before a specified date and time. If no value is entered, then the last modification date is applied. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`[1] Example: `${'2011-03-17T18:24:36.000+01:00'}`
`lastModifier`	A filtering value to limit the display of task flow content to folders and files last modified by a particular user. Enter the user name as specified by the user login credentials. Only one user name may be entered. If no value is entered, then all modified documents are shown. Example: `${'monty'}`

Table 31–8 (Cont.) Document List Viewer Task Flow Parameters

Task Flow Parameter	Description
`pageSize`	The maximum number of rows to show in the task flow. If the listing of folders and files in the task flow is larger than the specified number of rows, the task flow displays a scroll bar. Default: `27` Typical scenarios where you may wish to alter this value are: ■ The majority of end users have bigger screens, allowing for the display of more rows. ■ You want to fit this task flow into a smaller area of a page. **Note**: If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the task flow scroll bar with the application scroll bar.
`startFolderPath`	The name of the folder to use as the root folder in the current task flow instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the task flow instance. There is no need to set this value for task flows that display the content of the current space's default root folder. But it is useful, for example, when you want the start folder to be other than a space's default root folder and when you want to display content from another space. Example: ■ `${'/PersonalSpaces/monty/Public'}` ■ `${'/WebCenterB5/Proj_X/Specs'}` You can specify an EL expression to set this value. Default: The root folder of the content repository configured with the specified connection for the current Space.
`showFolders`	Specifies whether to show documents and folders, or documents only: ■ `${true}`: show folders and documents ■ `${false}` (default): show documents and hide folders
`taskFlowInstId`	The unique identifier for this task flow instance, used internally to maintain the association of the task flow instance with its customization and personalization settings and to manage saved queries. The value is generated automatically when you add individual files or the content of folders to your page from the Application Resources Panel, displaying the content in a Document List Viewer task flow. The value is blank when you add a Document List Viewer task flow onto your page from the Resource Palette. Enter or edit this value to set the unique identifier for the task flow.

[1] "TZ" is the time zone indicator. If the time being described is in UTC (Coordinated Universal Time), then the time zone indicator is "Z". If the time is from any other time zone, then TZ describes the offset from UTC of the time zone. For example, if the time is in California in December (Pacific Standard Time, PST), then the TZ indicator would be "-08:00".

31.7.4 Document Manager Task Flow Parameters

The Document Manager task flow displays folders and files as specified by its `layout` parameter: it may display folders and files in two panes (`explorer` layout), or a single pane showing only the content of the current folder (`table` layout), or a single pane showing the folder hierarchy starting from the root folder (`treeTable` layout). The `explorer` layout is identical to the Document Explorer task flow, without the properties `showDocuments`, `showFolders`, and `treeNavCollapsed`.

The Document Manager task flow provides comprehensive document management functionality, such as copying, moving, pasting, and deleting folders and files.

See Also:

- "Understanding the Document Manager Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Manager task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–9 describes the Document Manager task flow parameters.

Table 31–9 Document Manager Task Flow Parameters

Task Flow Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`layout`	A target layout for the task flow. Select from: - `${'explorer'}` (default): Displays folders and files in two panes; the left pane shows folders, and the right pane show the contents of the currently selected folder. This layout looks identical to the Document Explorer task flow, without the parameters `showDocuments`, `showFolders`, and `treeNavCollapsed`. See Figure 31–3. - `${'table'}`: Displays only the contents of the current folder in a single pane, with the capability to click a folder to drill down, refreshing the pane with the folder contents. See Figure 31–4. - `${'treeTable'}`: Displays the folder hierarchy in a single pane, beginning with the root folder, with the capability to expand and collapse folders. See Figure 31–5.

Table 31–9 (Cont.) Document Manager Task Flow Parameters

Task Flow Parameter	Description
`pageSize`	The maximum number of rows to show in the task flow. If the listing of folders and files in the task flow is larger than the specified number of rows, the task flow displays a scroll bar. Default: `27` Typical scenarios where you may wish to alter this value are: ■ The majority of end users have bigger screens, allowing for the display of more rows. ■ You want to fit this task flow into a smaller area of a page. **Note**: If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the task flow scroll bar with the application scroll bar.
`readOnly`	Specifies whether to disable and hide all content management operations: ■ `${true}`: Disable content management. ■ `${false}` (default): Expose content management to users.
`resourceId`	The currently focused resource. This value can be a folder ID or a document ID. The `resourceId` value is checked for coherence with the `connectionName` and `startFolderPath` values, and influences their default values when they are not explicitly specified. For example, if `resourceId` alone is specified, the `connectionName` value will be inherited from it and the startFolderPath will be either the resource itself for a folder or the parent folder for a file.
`startFolderPath`	The name of the folder to use as the root folder in the current task flow instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the task flow instance. There is no need to set this value for task flows that display the content of the current space's default root folder. But it is useful, for example, when you want the start folder to be other than a space's default root folder and when you want to display content from another space. Example: ■ `${'/PersonalSpaces/monty/Public'}` ■ `${'/WebCenterB5/Proj_X/Specs'}` You can specify an EL expression to set this value. Default: The root folder of the content repository configured with the specified connection for the current Space.

31.7.5 Document Navigator Task Flow Parameters

The Document Navigator task flow displays folders and files in a single pane, with the capability to expand and collapse folders to view folder hierarchy within the current folder. There are no menu options available to the end user for this task flow.

See Also:

- "Understanding the Document Navigator Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Navigator task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–10 describes the Document Navigator task flow parameters.

Table 31–10 Document Navigator Task Flow Parameters

Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`pageSize`	The maximum number of rows to show in the task flow. If the listing of folders and files in the task flow is larger than the specified number of rows, the task flow displays a scroll bar. Default: `27` Typical scenarios where you may wish to alter this value are: ■ The majority of end users have bigger screens, allowing for the display of more rows. ■ You want to fit this task flow into a smaller area of a page. **Note**: If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the task flow scroll bar with the application scroll bar.
`readOnly`	Specifies whether to disable and hide all content management operations: ■ `${true}`: Disable content management. ■ `${false}` (default): Expose content management to users.
`resourceId`	The currently focused resource. This value can be a folder ID or a document ID. The `resourceId` value is checked for coherence with the `connectionName` and `startFolderPath` values, and influences their default values when they are not explicitly specified. For example, if `resourceId` alone is specified, the `connectionName` value will be inherited from it and the startFolderPath will be either the resource itself for a folder or the parent folder for a file.
`showDocuments`	Specifies whether the navigation tree shows documents and folders, or folders only: ■ ${true} (default): Show documents and folders. ■ ${false}: Show folders and hide documents.

Table 31–10 (Cont.) Document Navigator Task Flow Parameters

Parameter	Description
`startFolderPath`	The name of the folder to use as the root folder in the current task flow instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the task flow instance. There is no need to set this value for task flows that display the content of the current space's default root folder. But it is useful, for example, when you want the start folder to be other than a space's default root folder and when you want to display content from another space. Example: ■ `${'/PersonalSpaces/monty/Public'}` ■ `${'/WebCenterB5/Proj_X/Specs'}` You can specify an EL expression to set this value. Default: The root folder of the content repository configured with the specified connection for the current Space.

31.7.6 Folder Viewer Task Flow Parameters

The Folder Viewer task flow displays a listing of the contents of a folder in a single pane as a flat listing.

See Also:

- "Understanding the Folder Viewer Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Folder Viewer task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–11 describes the Folder Viewer task flow parameters.

Table 31–11 Folder Viewer Task Flow Parameters

Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.

Table 31–11 (Cont.) Folder Viewer Task Flow Parameters

Parameter	Description
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`pageSize`	The maximum number of rows to show in the task flow. If the listing of folders and files in the task flow is larger than the specified number of rows, the task flow displays a scroll bar. Default: `27` Typical scenarios where you may wish to alter this value are: ■ The majority of end users have bigger screens, allowing for the display of more rows. ■ You want to fit this task flow into a smaller area of a page. **Note**: If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the task flow scroll bar with the application scroll bar.
`readOnly`	Specifies whether to disable and hide all content management operations: ■ `${true}`: Disable content management. ■ `${false}` (default): Expose content management to users.
`resourceId`	The currently focused resource. This value can be a folder ID or a document ID. The `resourceId` value is checked for coherence with the `connectionName` and `startFolderPath` values, and influences their default values when they are not explicitly specified. For example, if `resourceId` alone is specified, the `connectionName` value will be inherited from it and the startFolderPath will be either the resource itself for a folder or the parent folder for a file.
`showFolders`	Specifies whether the navigation tree shows documents and folders, or documents only: ■ `${true}` (default): Show folders and documents. ■ `${false}`: Show documents and hide folders.

Table 31–11 (Cont.) Folder Viewer Task Flow Parameters

Parameter	Description
`startFolderPath`	The name of the folder to use as the root folder in the current task flow instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the task flow instance. There is no need to set this value for task flows that display the content of the current space's default root folder. But it is useful, for example, when you want the start folder to be other than a space's default root folder and when you want to display content from another space. Example: ■ `${'/PersonalSpaces/monty/Public'}` ■ `${'/WebCenterB5/Proj_X/Specs'}` You can specify an EL expression to set this value. See "EL Expressions Relevant to the Documents Service" in Appendix B in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. Default: The root folder of the content repository configured with the specified connection for the current Space.

31.7.7 Recent Documents Task Flow Parameters

The Recent Documents task flow displays a listing of the most recently created or modified files by the current user.

See Also:

- "Understanding the Recent Documents Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Recent Documents task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–12 describes the Recent Documents task flow parameters.

Table 31–12 Recent Documents Task Flow Parameters

Parameter	Description
`connectionName`	The name of the content repository connection. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see "Registering Content Repositories" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.

Table 31–12 (Cont.) Recent Documents Task Flow Parameters

Parameter	Description
`groupSpace`	(Used in WebCenter Spaces only) The name of a space that is the source of the recently created or modified documents listed in the task flow. Valid values are: ■ no value or `${null}` (default): Displays documents for the entire connected content repository. ■ Display name or ID of a specific Space: Displays documents in that Space. ■ `${'all'}`: Displays documents from any Space, excluding non-Space documents.
`lastModifiedAfter`	A filtering value to limit the display of task flow content to folders and files last modified after a specified date and time. If no value is entered, then content modified in the last three months is shown. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`[1] Example: `${'2011-03-17T18:24:36.000+01.00'}`
`lastModifiedBefore`	A filtering value to limit the display of task flow content to folders and files modified before a specified date and time. If no value is entered, then the last modification date is applied. The value uses the ISO 8601 format: `${'yyyy-mm-ddThh:mm:ss.sssTZD'}`[1] Example: `${'2011-03-17T18:24:36.000+01:00'}`
`lastModifier`	A filtering value to limit the display of task flow content to folders and files last modified by a particular user. Enter the user name as specified by the user login credentials. Only one user name may be entered. If no value is entered, then all modified documents are shown. Example: `${'monty'}`
`maxDocuments`	The maximum number of files to show. If no value or 0 is entered, then up to 10 of the most recently accessed documents are shown. Example: `${10}` Note that there is no single quote surrounding the value. Default: `${null}`
`mostRecentFirst`	Specifies the sort order of files in the task flow: ■ `${true}` (default): Most recent documents listed first. ■ `${false}`: Oldest documents listed first.

[1] "TZ" is the time zone indicator. If the time being described is in UTC (Coordinated Universal Time), then the time zone indicator is "Z". If the time is from any other time zone, then TZ describes the offset from UTC of the time zone. For example, if the time is in California in December (Pacific Standard Time, PST), then the TZ indicator would be "-08:00".

31.7.8 Document Viewer Task Flow Parameters

The Document Viewer task flow displays the contents of a folder or a preview of an individual file in the default template for its file type.

See Also:

- "Understanding the Document Viewer Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Viewer task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–13 describes the Document Viewer task flow parameters.

Table 31–13 Document Viewer Task Flow Parameters

Parameter	Description
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`initialSidebar`	Specifies the tabbed pane to have initial focus in the Document Viewer. Valid values are: ▪ `${'comments'}` (default). Displays the **Comments** pane (see "Viewing, Entering, and Deleting Comments on a File" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*). ▪ `${'tags'}`. Displays the **Tags** pane (see "Working with Tags" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*). ▪ `${'history'}`. Displays the **History** pane (see "Viewing and Deleting File Version History" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*). ▪ `${'docInfo'}`. Displays the **Info** pane (see "Viewing and Modifying Folder and File Properties" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*). ▪ `${'tags'}`. Displays the **Links** pane (see "Working with Links" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*). ▪ `${'autovue'}`. When Oracle AutoVue is installed, displays an **AutoVue** pane showing the AutoVue markup for the current document in a table of hyperlinked markup names (see "Collaborating on Documents Using AutoVue" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*).
`readOnly`	Specifies whether to disable and hide all content management operations: ▪ `${true}`: Disable content management. ▪ `${false}` (default): Expose content management to users.
`resourceID`	The currently focused resource. This value can be a folder ID or a document ID.

31.7.9 Document Mini Properties Task Flow Parameters

The Document Mini Properties task flow displays the basic properties of a selected file in a read-only view. This task flow is available for all file types.

See Also:

- "Understanding the Document Mini Properties Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Mini Properties task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–14 describes the Document Mini Properties task flow parameters.

Table 31–14 Document Mini Properties Task Flow Parameters

Parameter	Description
`resourceID`	The ID of the document for which to display basic properties.

31.7.10 Document Properties Task Flow Parameters

The Document Properties task flow displays both Basic and Advanced properties of a selected file, along with an **Edit** button to modify property values. This choice is available for all file types.

See Also:

- "Understanding the Document Properties Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Properties task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–15 describes the Document Properties task flow parameters.

Table 31–15 Document Properties Task Flow Parameters

Parameter	Description
`resourceID`	The ID of the document for which to display basic and advanced properties.
`readOnly`	Specifies whether or not to allow end user to edit document properties: ▪ `${true}`: Do not display **Edit** button in Basic and Advanced properties panes, thus disabling the ability for the end user to edit document properties. ▪ `${false}` (default): Display **Edit** button in Basic and Advanced properties panes, allowing end users to edit document properties.

31.7.11 Rich Text Editor Task Flow Parameters

The Rich Text Editor task flow displays an HTML or wiki document in the Rich Text Editor.

See Also:

- "Using the Rich Text Editor (RTE)" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Rich Text Editor task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–18 describes the Rich Text Editor task flow parameters.

Table 31–16 Rich Text Editor Task Flow Parameters

Parameter	Description
`destinationID`	The currently focused resource. This value can be a folder ID to create a new document in that folder, or a document ID (as shown in the properties pane of the Document Viewer task flow) to edit an existing document.
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`
`initialSidebar`	Specifies a pane to display in the sidebar of the Document Viewer. Valid values are: ▪ `${'comments'}` (default). Displays a **Comments** pane (see "Viewing, Entering, and Deleting Comments on a File" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'history'}`. Displays a **History** pane (see "Viewing and Deleting File Version History" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'tags'}`. Displays a **Tags** pane (see "Working with Tags" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'links'}`. Displays a **Links** pane (see "Working with Links" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'recommendations'}`. Displays a **Recommendations** pane (see "Working with Recommendations" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'relatedItems'}`. Displays a **Related Items** pane (see "Working with Related Items (Links, Tags, and Recommendations)" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*) ▪ `${'docInfo'}`. Displays a **Properties** pane (see "Viewing and Modifying Folder and File Properties" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*)

31.7.12 Document Upload Task Flow Parameters

The Document Upload task flow displays the Upload dialog to allow users to upload new documents to the page.

See Also:

- "Uploading New Files" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Upload task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–18 describes the Document Upload task flow parameters.

Table 31–17 Document Upload Task Flow Parameters

Parameter	Description
`destinationID`	The currently focused resource. This value specifies a folder ID to upload a document into that folder.
`destinationSelectionEnabled`	Specifies whether or not the destination folder can be changed: ■ `${true}`: Display destination folder as an input text box along with a **Browse** button. ■ `${false}` (default): Display destination folder as read only text.
`featuresOff`	A list of disabled features for the task flow. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin`, `checkout`, `clipboard`, `close`, `delete`, `download`, `dnd`, `editwiki`, `editoffice`, `newfolder`, `newwiki`, `rename`, `upload`, `multifile-upload`, `profile-upload`, `search`, `advancedSearch`, `workflow`, `properties`, `history`, `comments`, `likes`, `links`, `tags`, `recommendations`, `autovue`, `title`, `related-items`, `social`, `sidebars`, `ils`. Example: `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`

31.7.13 Document Version History Task Flow Parameters

The Document Version History task flow displays a list of versions of a selected file in a read-only view. This task flow is available for all file types.

See Also:

- "Understanding the Document Version History Task Flow" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the Document Version History task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–18 describes the Document Version History task flow parameters.

Table 31–18 Version History Task Flow Parameters

Parameter	Description
`horizontalLayout`	Specifies the layout orientation for the version history information: ■ `${horizontal}`: Horizontal orientation. ■ `${vertical}` (default): Vertical orientation.
`resourceID`	The ID of the document for which to display version history.
`readOnly`	Specifies whether to disable and hide all content management operations: ■ `${true}`: Disable content management. `${false}` (default): Expose content management to users.

31.7.14 AutoVue Task Flow Parameters

The AutoVue task flow integrates the Oracle AutoVue functionality to allow users to review and collaborate on a document.

See Also:

- "Collaborating on Documents Using Oracle AutoVue" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*
- Section 31.5, "Adding a Content Task Flow to a Page"

Parameters that are unique to the AutoVue task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 31–19 describes the AutoVue task flow parameters.

Table 31–19 AutoVue Task Flow Parameters

Parameter	Description
`resourceId`	The ID of the document to display in the AutoVue task flow.

32

Integrating Wikis and Blogs

You can integrate the wiki and blog functionality in your WebCenter Portal: Framework applications to enable your application users to collaborate and share content and ideas.

This chapter describes how to integrate the wiki and blog functionality into your Framework applications. It includes the following sections:

- Section 32.1, "Introduction to Wikis and Blogs"
- Section 32.2, "Requirements for Wikis and Blogs"
- Section 32.3, "Integrating Wikis"
- Section 32.4, "Integrating Blogs"

32.1 Introduction to Wikis and Blogs

A wiki is a collection of useful content or information that users can browse, update, and remove, sometimes without the need for registration. This ease of interaction and the variety of operations make wiki an effective tool for collaborative authoring, where multiple people create written content together.

Blogs are typically personal records of an individual user's experience and opinions. The word *blog* is a contraction of the term *Web log*. It was coined to describe the online diaries spawned in the late 1990s. Blogs provide a useful tool for discussing and/or evangelizing any type of idea, strategy, or point of view. Blogs may be projected out to a select group of people or to a wider audience. Typically, each blog contains various blog posts, with the most recently added blog post displayed at the top. Blogs invite readers to comment on the overall concepts.

You can integrate wikis and blogs into your Framework applications using the Documents service. There is no limitation on the number of wiki pages or blogs that you can expose in an application.

32.2 Requirements for Wikis and Blogs

To support the wiki and blog functionality, the Documents service relies on Content Server. For wiki and blog functionality to be available in your Framework applications, the following prerequisites must be met:

- Content Server 11*g* must be installed and configured as the default content repository for WebCenter Portal. The wiki blog functionality is not available with Content Server 10*g*. For information about installing and configuring Content Server 11*g*, see the "Managing Content Repositories" chapter in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

- A connection to Content Server 11*g* must be established. For information, see Section 27.2.1, "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter."
- Complete the steps covered in Section 7.2, "Preparing Your Framework Application to Consume Services", as needed. For example, you can use the Activity Stream feature to track activities such as when a blog or wiki document is created or updated. Application users can add comments on wiki documents and blog posts to express their views. To enable the Comments and Activities Stream features, you must register a connection to a database that has the `WebCenter` schema installed.

After completing these requirements, you can use the Documents service to add wikis and blogs to your Framework applications:

- To add wikis to your application, you will need to add a Documents service task flow that provides the **Add Wiki Document** action (such as the Document Explorer task flow). See Section 32.3, "Integrating Wikis."
- To add blogs to your application, you use the blog tasks flows. See Section 32.4, "Integrating Blogs."

If you want to convert wikis and blogs in your Framework applications into PDF, your application administrator must configure the WebCenter Conversion component, as described in "Enabling the Conversion of Wikis and Blogs into PDFs" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

32.3 Integrating Wikis

This following sections provide the information you need to add wiki documents to your Framework applications:

- Section 32.3.1, "What You Should Know About the Document Explorer Task Flow"
- Section 32.3.2, "Adding the Document Explorer Task Flow to Provide the Wiki Functionality"
- Section 32.3.3, "Modifying Document Explorer Task Flow Parameters"
- Section 32.3.4, "What Happens at Runtime"
- Section 32.3.5, "Displaying Wiki Page Links Within Content Presenter"

32.3.1 What You Should Know About the Document Explorer Task Flow

To provide your application users with the ability to add and manage wiki documents at runtime, you need to add the Document Explorer task flow to your Framework application. When you add this task flow at design time, it is built into the application for end users, and enables them to work with wiki documents at runtime.

The Document Explorer task flow has associated parameters. For information, see Section 31.7.2, "Document Explorer Task Flow Parameters."

32.3.2 Adding the Document Explorer Task Flow to Provide the Wiki Functionality

To prepare your Framework application and add the Document Explorer task flow for enabling the wiki functionality:

1. Ensure that you have completed the requirements listed in Section 32.2, "Requirements for Wikis and Blogs."

Note: While setting up the connection, you must choose an authentication method. If you choose identity propagation, you must implement ADF security in your application and configure secure socket layer (SSL) on Oracle Content Server.

If you choose external application, you must specify the external application you want to use for authenticating users to Oracle Content Server.

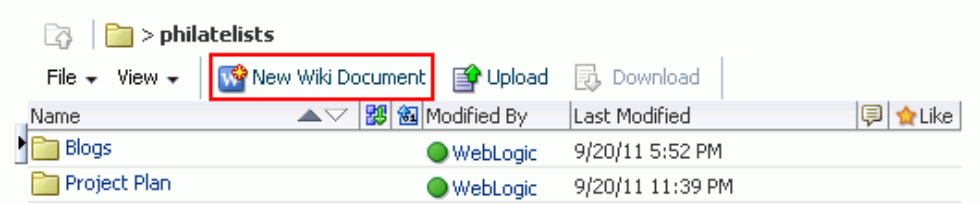
For information, see Section 7.2.1.1, "Implementing Security for Services."

2. Create or open the JSF page in which you want to provide the wiki functionality (see Section 5.3, "Adding Pages to a Portal").
3. Choose a method for adding the Document Explorer task flow to the page:
   - To add the Document Explorer task flow prepopulated with a selected folder or file, see Section 31.4, "Adding a Selected Folder or File to a Page"
   - To add the Document Explorer task flow standalone as an "empty" task flow that you can populate using the task flow menus and actions, see Section 31.5, "Adding a Content Task Flow to a Page"

   For information about the parameters that you can set for the Document Explorer task flow, see Section 31.7.2, "Document Explorer Task Flow Parameters."

   When you run the page, the Document Explorer task flow displays in your default browser, as shown in Figure 32–1. Application users can create a wiki document by clicking **New Wiki Document**.

Figure 32–1 Creating a New Wiki Document



32.3.3 Modifying Document Explorer Task Flow Parameters

The way you modify parameters of a Document Explorer task flow is same as for any other Documents service task flow. For information, see Section 31.6, "Modifying Content Task Flow Parameters." For information about the specific parameters that you can modify, see Section 31.7.2, "Document Explorer Task Flow Parameters."

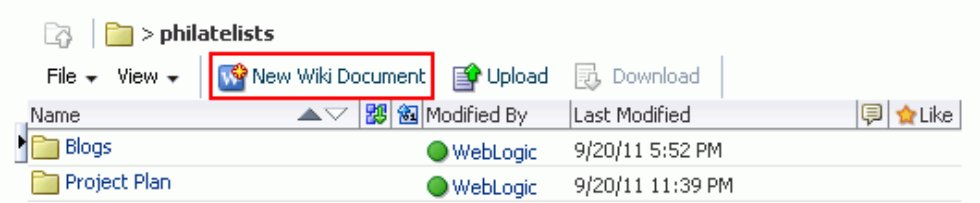
32.3.4 What Happens at Runtime

At runtime, authorized application users can create wiki documents in a Framework application in which the wiki functionality has been exposed. Users can also edit, download, rename, share, and subscribe to a wiki document and view its history and properties.

At runtime, the folder that you exposed through the Document Explorer task flow is shown as the root folder. Application users can create a wiki document by clicking

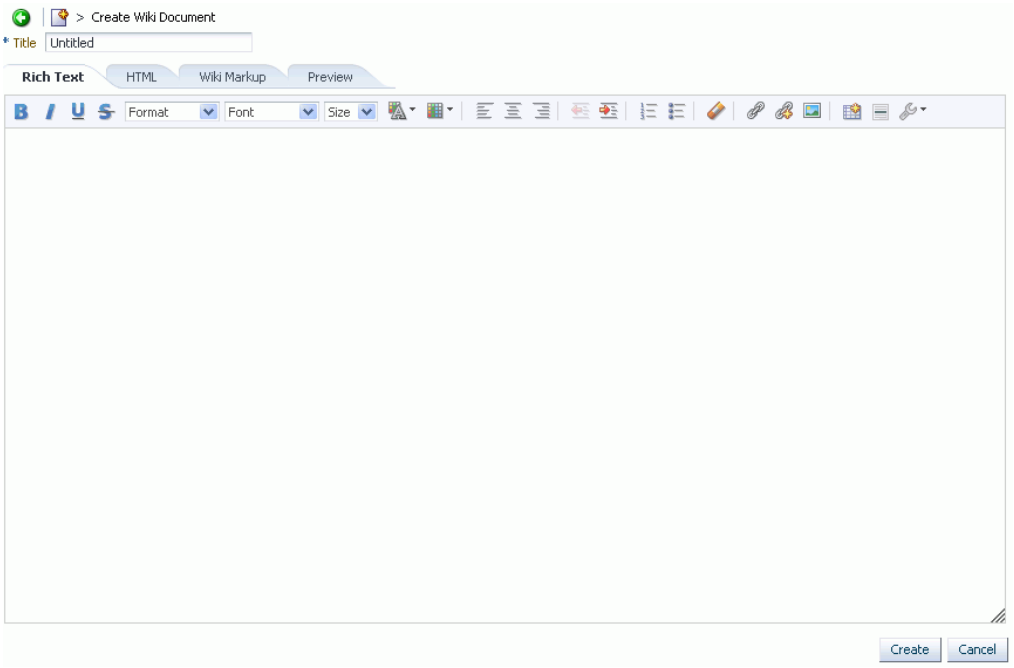
New Wiki Document (Figure 32–4), which opens the Rich Text Editor (RTE) (Figure 32–3), displaying a blank wiki document.

Figure 32–2 Creating a New Wiki Document



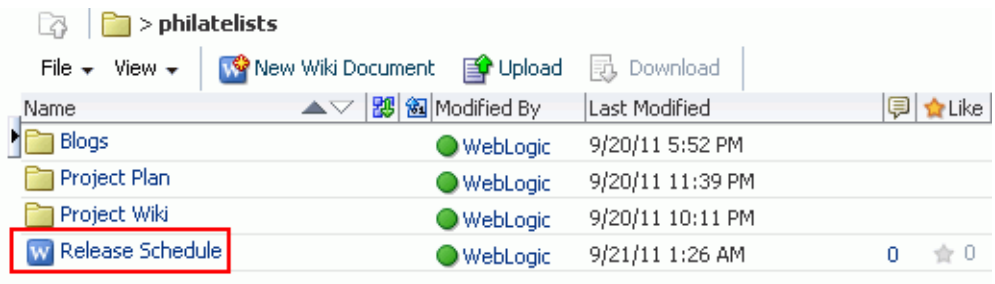
Users can add and edit wiki content using the Rich Text Editor (RTE), which is a fully-integrated HTML text editor (Figure 32–3).

Figure 32–3 Rich Text Editor Displaying New Wiki Document



By default, all wiki documents are created directly under the root folder (Figure 32–4).

Figure 32–4 Wiki Document Created Using New Wiki Document Action



At runtime, users can also choose to display a wiki document using Content Presenter.

The runtime procedures for adding and managing wiki documents in a Framework application are same as in WebCenter Portal: Spaces. For generic information, refer to the chapter "Working with Wiki Documents" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

32.3.5 Displaying Wiki Page Links Within Content Presenter

At runtime, when a wiki document is displayed using a Content Presenter display template, by default the links present in the wiki document render in the Document Viewer. You can configure your application to display such links within Content Presenter.

To open the links present in a wiki document within Content Presenter:

1. Open the `adf-config.xml` file. This file is located under `Descriptors/ADF META-INF` in the Application Navigator.
2. Update the `Namespace` entry as follows:

```
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
```

3. Update the `ResourceActionHandler` entry as follows:

```
<wpsC:adf-service-config
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
      <resource-handler
class="oracle.webcenter.portalframework.sitestructure.handler.NavigationResourc
eActionHandler"/>
</wpsC:adf-service-config>
```

4. Save `adf-config.xml`.

32.4 Integrating Blogs

When you integrate the blogs functionality into your Framework application, authorized application users can create and manage blog posts. To enable the blog functionality, you need to add the blog task flows.

While integrating the Blogs task flow, you need to specify a folder on Oracle Content Server that will act as the container for your blog and contain all the blog posts.

This following sections provide the information you need to add blogs to a Framework application:

- Section 32.4.1, "Understanding the Blog Task Flows"
- Section 32.4.2, "Adding a Blog to a Page"
- Section 32.4.5, "Blog Task Flow Parameters"
- Section 32.4.4, "Modifying Blog Task Flow Parameters"
- Section 32.4.3, "What Happens at Runtime"

32.4.1 Understanding the Blog Task Flows

If you want to include a blog on a page, along with other page components, you can use the blog task flows to add one or more elements of a blog to a page. Table 32–1 lists and describes the blog task flows:

Table 32–1 Blog Task Flows

Blog Task Flow	Description	Example as Exposed on a Page
Blog Archives	Displays a composite list of blogs based on dates.	
Blog Banner	Displays a banner for the blog.	
Blog Digest	Displays a blog or blog post.	
Recent Blog Posts	Displays a list of most recent blog posts.	

Table 32–1 (Cont.) Blog Task Flows

Blog Task Flow	Description	Example as Exposed on a Page
Blogs	Displays a blog or blog post with a default design.	

32.4.2 Adding a Blog to a Page

To add a blog to a page in your Framework application:

1. Ensure that you have completed the requirements listed in Section 32.2, "Requirements for Wikis and Blogs."

   Note: While setting up the connection, you must choose an authentication method. If you choose identity propagation, you must implement ADF security in your application and configure secure socket layer (SSL) on Oracle Content Server.

   If you choose external application, you must specify the external application you want to use for authenticating users to Oracle Content Server.

   For information, see Section 7.2.1.1, "Implementing Security for Services."

2. Create or open a page in your application where you want to expose the blog functionality (see Section 5.3, "Adding Pages to a Portal").
3. Before you add a blog task flow to your page, consider whether or not you want to allow your application users to edit the task flow. To enable runtime editing of the page, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."
4. Add a blog to the page using an existing folder, or by creating a new "empty" blog:
   - Section 32.4.2.1, "Establishing a Selected Folder as a Blog Folder"
   - Section 32.4.2.2, "Adding a Custom Blog to a Page Using the Blog Task Flows"

32.4.2.1 Establishing a Selected Folder as a Blog Folder

If you have an existing folder, perhaps already containing blog posts, you can establish that folder as a blog folder, instead of creating a new "empty" blog (see Section 32.4.2.2, "Adding a Custom Blog to a Page Using the Blog Task Flows"). At runtime, the blog displays the blog posts in that folder.

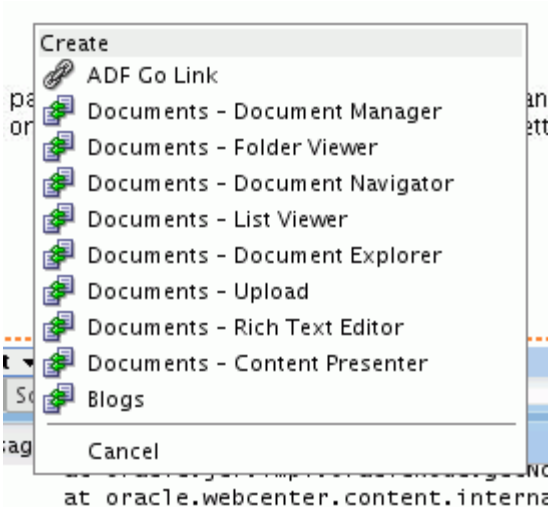
To expose a selected folder and its contents as a blog on a page:

1. In the **Application Resources** panel, navigate to **Connections**, then **Content Repository**.
2. Locate the name of your Content Server connection, then expand it to navigate to the folder that you want to establish as the blog folder.

   > **Tip:** You can expose any folder as a blog. To better organize your blogs, you may consider storing each blog folder in a parent folder; for example, a folder named **Blogs**. You can create a folder using either the Documents service interface or the Content Server console.
   >
   > Files stored in the blog folder are identified as blog posts in their metadata `Type` field. Uploading an HTML file into a blog folder does not expose it as as blog post, unless its `Type` is set to `blog`.

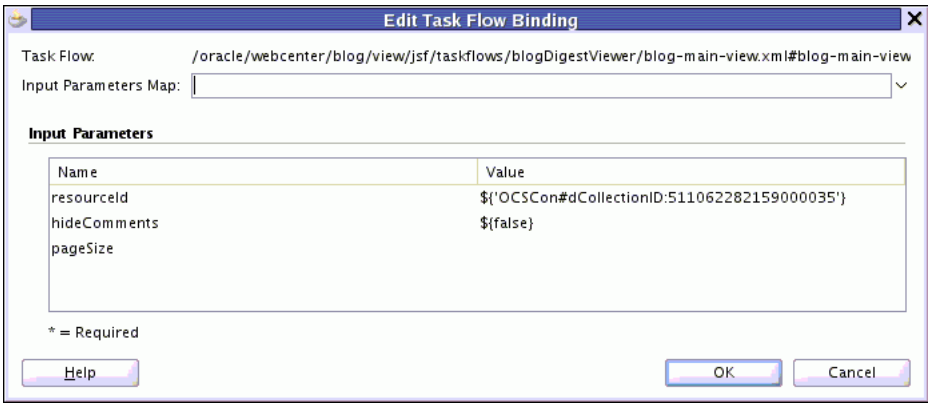
3. Drag the folder onto the page, and release the mouse button to display a menu listing all available task flows (Figure 32–5).

Figure 32–5 Menu of Available Task Flows for Selected Folder



4. From the menu, choose **Blogs** to open the Edit Task Flow Binding dialog (Figure 32–6).

Figure 32–6 Specifying Parameters for Blogs Task Flow



5. In the Edit Task Flow Binding dialog, the `resourceId` value is automatically populated with the resource ID of the selected folder. Click **OK** to accept the default task flow parameters, or modify the parameters as desired. For more information, see Section 32.4.5.5, "Blogs Task Flow Parameters."

6. Save your page and the page definition file, then run your page to your browser by right-clicking the page (not the page definition file) and choosing **Run**.

The new blog is given the same name as the selected folder. Existing files under the folder become blog posts if the `Type` property of the file is set to `blog` (for more information about file properties in Content Server, see "Info Update Form" in *Oracle WebCenter Content User's Guide for Content Server*). As posts are added to the blog at runtime, the blog posts are stored in this folder.

32.4.2.2 Adding a Custom Blog to a Page Using the Blog Task Flows

You may want to add an "empty" blog task flow onto your page, either as one component on a page, or as the entire page. For example, in cases where you do not have a content repository connection, or want to use an EL expression as a parameter to dynamically bind to a target item, you may want to use this method instead of the method described in Section 32.4.2.1, "Establishing a Selected Folder as a Blog Folder." Additionally, the individual blog task flows enable you to customize the appearance of a blog.

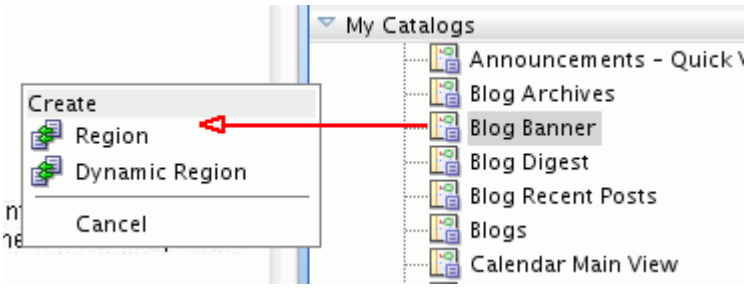
To add a customized blog task flow to a page:

1. In the **Resource Palette**, expand the **WebCenter Portal - Services Catalog**. Under **Task Flows**, select the blog task flow that you want to add, drag it onto your page., and realease the mouse button.

   For information about each of the five blog task flows, see Section 32.4.1, "Understanding the Blog Task Flows."

2. From the menu that displays, choose **Region** (Figure 32–7).

Figure 32–7 Adding Blog Task Flow Using Resource Palette



3. In the Edit Task Flow Binding dialog, click **OK** to accept the default task flow parameters, or modify the parameters as desired. For information, see Section 32.4.5, "Blog Task Flow Parameters."
4. Repeat these steps for each blog task flow that you want to comprise your blog.
5. Save your page and the page definition file, then run your page to your browser by right-clicking the page (not the page definition file) and choosing **Run**.

Internally, the new blog is given the same name as the folder specified by the `resourceID`; however, the name visible on the page is the value you specify in the `Title` parameter of the Blog Banner task flow. Existing files under the folder become blog posts if the `Type` property of the file is set to `blog` (for more information about file properties in Content Server, see "Info Update Form" in *Oracle WebCenter Content User's Guide for Content Server*). As posts are added to the blog at runtime, the blog posts are stored in this folder.

32.4.3 What Happens at Runtime

At runtime, the folder specified by the `resourceID` parameter for the blog task flow is exposed as a blog on the page:

- If you created a blog as described in Section 32.4.2.1, "Establishing a Selected Folder as a Blog Folder", then the name of the folder that you exposed as a blog is displayed as the name of your blog in the blog banner.
- If you created a blog as described in Section 32.4.2.2, "Adding a Custom Blog to a Page Using the Blog Task Flows", then the name of the folder that you specify in the `resourceID` parameter of the blog task flow is the name of your blog internally; however, the name visible on the page is the value you specify in the `Title` parameter of the Blog Banner task flow.

Authorized users can create blog posts, and the posts are displayed on the blog page. Users can perform tasks such as edit, download, comment upon blog posts, and so on.

On the default blog page (Figure 32–8), and in the Blog Archives task flow, the **Archives** section provides links to blog posts by year and by month. Clicking a month displays all blog posts created during that month.

Figure 32–8 A Default Blog Page



A blog displays various details for each blog post. These include the blog post title, the blog post content, date of creation or modification, name of the user who created or last modified the post, and the number of comments on the blog post. Authorized users can click the **Edit Post** icon to edit a post, and click the **Post comments** link to enter comments on a blog post.

Clicking a blog post title opens the blog post to occupy the entire the blog page (Figure 32–9), providing controls to manage the blog post.

Figure 32–9 A Blog Post



The procedures for managing blogs at runtime in a Framework application is same as that for managing blogs in WebCenter Portal: Spaces. For information, see the "Working with Blogs" chapter in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

32.4.4 Modifying Blog Task Flow Parameters

The procedure for modifying the parameters of a Blogs task flow is same as any other Documents service task flow. For information, see Section 31.6, "Modifying Content

Task Flow Parameters." For information about the parameters that you can modify, see Section 32.4.5, "Blog Task Flow Parameters."

32.4.5 Blog Task Flow Parameters

Each blog task flow has its own set of parameters. You can configure these values when you add the task flow to your page, or modify the values at any time .

The following sections describe the parameters for the blog task flows:

- Section 32.4.5.1, "Blog Archives Task Flow Parameters"
- Section 32.4.5.2, "Blog Banner Task Flow Parameters"
- Section 32.4.5.3, "Blog Digest Task Flow Parameters"
- Section 32.4.5.4, "Blog Recent Posts Task Flow Parameters"
- Section 32.4.5.5, "Blogs Task Flow Parameters"

32.4.5.1 Blog Archives Task Flow Parameters

The Blog Archives task flow displays a composite list of blogs based on dates.

See Also: Section 32.4.1, "Understanding the Blog Task Flows."

Parameters that are unique to the Blog Archives task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 32–2 describes the Blog Archives task flow parameters.

Table 32–2 Blog Archives Task Flow Parameters

Parameter	Description
`resourceId`	The resource ID of the blog folder.

32.4.5.2 Blog Banner Task Flow Parameters

The Blog Banner task flow task flow displays a banner for the blog.

See Also: Section 32.4.1, "Understanding the Blog Task Flows."

Parameters that are unique to the Blog Banner task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 32–3 describes the Blog Banner task flow parameters.

Table 32–3 Blog Banner Task Flow Parameters

Parameter	Description
`resourceId`	The resource ID of the blog folder.
`imageURL`	(Optional) The background image to be used in the blog banner. When not specified, the background image will default to an image provided by the current skin.
`title`	(Optional) The title to be used for the blog banner. Default: The blog folder name.

32.4.5.3 Blog Digest Task Flow Parameters

The Blog Digest task flow displays a blog or blog post.

See Also: Section 32.4.1, "Understanding the Blog Task Flows."

Parameters that are unique to the Blog Digest task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 32–4 describes the Blog Digest task flow parameters.

Table 32–4 Blog Digest Task Flow Parameters

Parameter	Description
`resourceId`	The target blog resource to display. This can be either a folder ID, in which case the blog listing for this folder will display, or a document ID, in which case the blog post will display.
`year`	A Aour-digit number specifying the target year used to filter blog entries. Example: `2012`
`month`	A number from 1 to 12 specifying the target month used to filter blog entries. For this parameter to take effect, the Filter Year parameter must also be specified. Example: `10` (October)
`hideComments`	Specifies whether the Comments feature is exposed: ▪ Selected: Hide the Comments link and pane. ▪ Cleared (default): Show the Comments link and pane.
`pageSize`	The number of blog posts displayed in the Blog Digest Viewer before the Next and Previous icons are enabled. Default: `10`

32.4.5.4 Blog Recent Posts Task Flow Parameters

The Blog Recent Posts task flow displays a list of most recent blog posts.

See Also: Section 32.4.1, "Understanding the Blog Task Flows."

Parameters that are unique to the Blog Recent Posts task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 32–5 describes the Blog Recent Posts task flow parameters.

Table 32–5 Blog Recent Posts Task Flow Parameters

Parameter	Description
`resourceId`	The resource ID of the blog folder.
`recentPostListSize`	The number of recent posts to display. Default: `10`

32.4.5.5 Blogs Task Flow Parameters

The Blogs task flow displays a blog or blog post with a default design.

See Also: Section 32.4.1, "Understanding the Blog Task Flows."

Parameters that are unique to the Blogs task flow task flow are shown in the Edit Task Flow Binding dialog box when you add the task flow to a page. Table 32–6 describes the Blogs task flow task flow parameters.

Table 32–6 Blogs Task Flow Parameters

Parameter	Description
`resourceId`	The target blog resource to display. This can be either a folder ID, in which case the blog listing for this folder will display, or a document ID, in which case the blog post will display.
`hideComments`	Specifies whether the Comments feature is exposed: ■ Selected: Hide the Comments link and pane. ■ Cleared (default): Show the Comments link and pane.
`pageSize`	The number of blog posts displayed in the Blogs task flow before the Next and Previous icons are enabled. Default: `10`

Part V

Enabling Communication and Collaboration

Part V contains the following appendixes:

- Chapter 33, "Integrating the Announcements Service"
- Chapter 34, "Integrating the Discussions Service"
- Chapter 35, "Integrating the Instant Messaging and Presence Service"
- Chapter 36, "Integrating the Mail Service"
- Chapter 37, "Integrating the Polls Service"

33

Integrating the Announcements Service

This chapter explains how to integrate the Announcements service into a WebCenter Portal: Framework application at design time. For more information about managing and including announcements, see:

- "Managing the Announcements and Discussions Services" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Announcements Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 33.1, "Introduction to Announcements"
- Section 33.2, "Basic Configuration for the Announcements Service"
- Section 33.3, "Advanced Information for the Announcements Service"

33.1 Introduction to Announcements

The Announcements service lets you create and expose announcements on your application pages. Access to announcements boosts community participation, problem resolution, and knowledge sharing.

You can use the Links service to link announcements to other services, such as Events or Discussions. For example, suppose your company is announcing a new product, you can link from the announcement directly to a discussion forum, where potential customers can ask other customers about the product, or link to an instant messenger to speak directly with a customer service representative to purchase the product.

Announcements can show either forum-level announcements or global announcements (that is, system announcements not specific to a forum).

This section includes the following subsections:

- Section 33.1.1, "Understanding the Announcements Service"
- Section 33.1.2, "Requirements for Announcements"
- Section 33.1.3, "What Happens at Runtime"

33.1.1 Understanding the Announcements Service

With the Announcements service, you can do the following:

- Create announcements.
- Edit and delete existing announcement.

- Mail announcements.
- Personalize the Announcements view. You can select to display all announcements or only announcements sent today, this week, or this month. You can also view future, active, or expired announcements.
- Manage announcements, such as publishing announcements on a later date and specifying auto-expire on certain dates.

The Announcements service is integrated with many other services in WebCenter Portal, such as Activity Stream, RSS, and Instant Messaging and Presence, and you can link announcements to other services, such as Events and Discussions. For example, suppose your company is announcing a new product. You can link from the announcement directly to a discussion forum, where potential customers can ask other customers about the product, or link to an instant messenger where customers can chat with a customer service representative about the product. With WebCenter Portal live search, announcement titles are searchable, and with Oracle Secure Enterprise Search, announcement titles *and* text are searchable.

33.1.2 Requirements for Announcements

The Announcements service requires a discussions server. You must install and configure the discussions server that comes with Oracle Fusion Middleware.

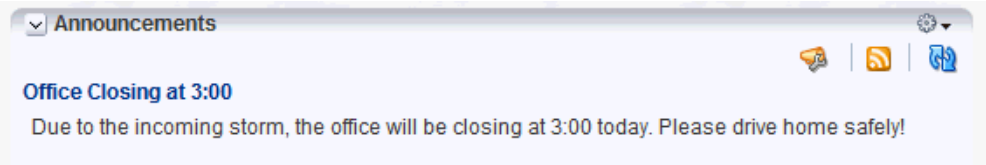
> **See Also:** *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*

33.1.3 What Happens at Runtime

At runtime, users who have been granted the `create` permission can post announcements, and these announcements are visible to all users who have `view` permission. For example, an application administrator can use this service to announce the availability of a new feature or the plan to shut down the application temporarily for maintenance.

Figure 33–1 shows a sample announcement at runtime.

Figure 33–1 Sample Announcement in the Announcements - Quick View Task Flow



To view announcements in the Announcement Manager, ensure that the Announcements task flow has been added to your Framework application. Then log in to your application and click the **Open Announcement Manager** icon in the Announcements task flow (Figure 33–2).

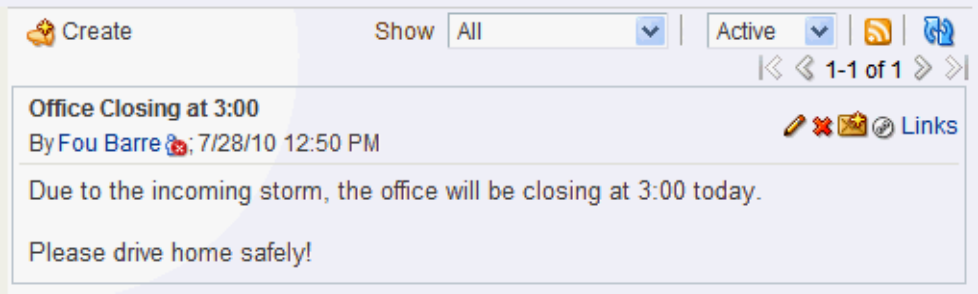
Figure 33–2 Open Announcement Manager Icon



Note: The **Open Announcement Manager** icon does not display if the user does not have the required privileges.

Figure 33–3 shows the sample announcement in the Announcement Manager.

Figure 33–3 Sample Announcement in the Announcement Manager



Application administrators in Framework applications can access the Announcement Manager, which provides **Create**, **RSS**, and **Refresh** icons, and **Show** dropdown lists to control the display of announcements on the page. **Edit**, **Delete**, **Mail**, and **Links** icons are available for each announcement in the list.

Depending on the privileges users have on the page and whether the required services in WebCenter Portal are configured in the application, users may see only a subset of these options in the Announcement Manager. For example, the **Delete** icon is displayed only to users with administrative privileges.

For more information about the service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

33.2 Basic Configuration for the Announcements Service

This section describes required steps for adding this service to your Framework application. It includes the following subsections:

- Section 33.2.1, "Setting Up a Connection for Announcements"
- Section 33.2.2, "Adding the Announcements Service at Design Time"
- Section 33.2.3, "Setting Security for Announcements"

33.2.1 Setting Up a Connection for Announcements

To take advantage of the Announcements service, you must first create a connection to the discussions server from your application. To do so, ensure that you have the connection information for the discussions server.

33.2.1.1 Announcements Connections

The Announcements service requires a Discussion Forum connection to the discussions server. You can register additional Discussion Forum connections, but only one connection is active at a time.

When you create a Discussion Forum connection or set a connection as active, both the Announcements and Discussions services use this same connection. If you have an existing connection, then you can skip this section and see Section 33.2.2, "Adding the Announcements Service at Design Time."

If you do not have an existing connection, then you must create a new Discussion Forum connection.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

33.2.1.2 How to Set Up Connections for Announcements

To set up the Announcements connection:

1. In Oracle JDeveloper, open the application in which you plan to consume Announcements.

   > **Note:** If you created a Discussion Forum connection for the Discussions service, then that is used by default for the Announcements service. No extra configuration is required.

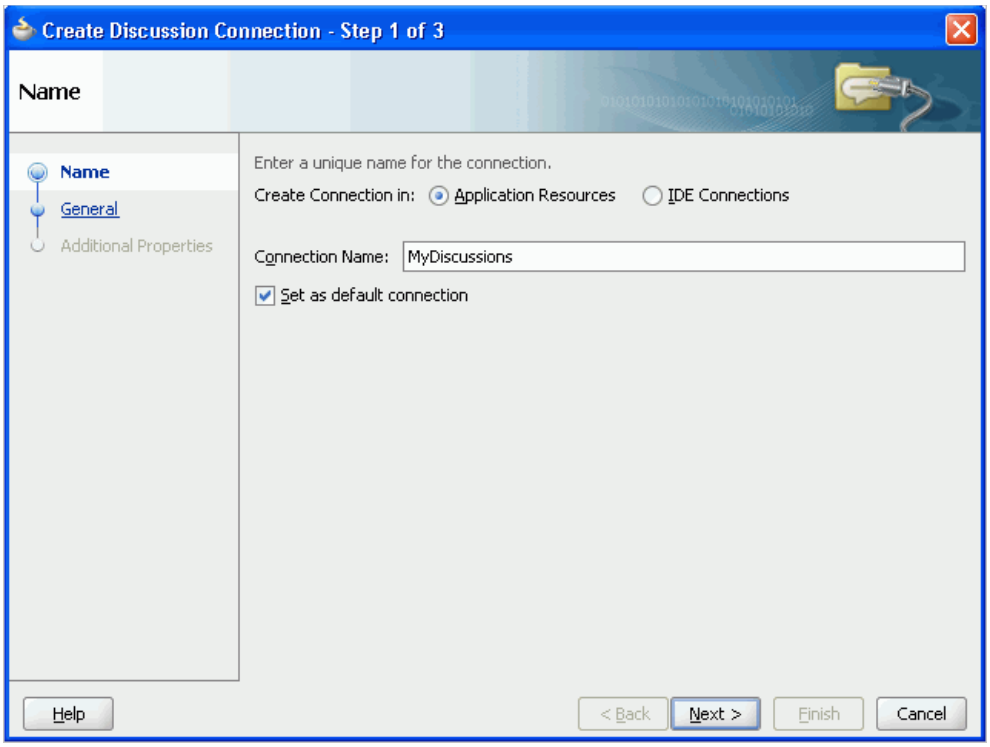
2. In the Application Navigator, under Application Resources, right-click **Connections**, then select the new connection **Discussion Forum** from the list.
3. On the **Name** page, select to create the connection in **Application Resources**. (A connection in Application Resources is available only for that application, while a connection in IDE Connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE Connections so you do not need to re-create it.)

   > **Note:** If you create a connection in IDE and not in the application, then the connection must be added to the application. For example, in the Resource Palette under IDE Connections, right-click the connection and select **Add to Application**.

4. For **Connection Name**, enter a meaningful name; for example, `MyDiscussions`.
5. Select the **Set as default connection** checkbox. You can have multiple connections, but only one can be active (default). If you have different discussions servers (for example, for each page in a space), then do not select the checkbox, but the service requires that one connection be marked as the active connection (Figure 33–4).

   > **Note:** After you create a connection as the active connection, you cannot edit it so that it is *not* the default. To use a different default connection, you must create a new connection and mark that as the default connection.

Figure 33–4 Create Discussion Connections, Step 1



6. Click **Next**.
7. On the **General** page, enter values for the required parameters:
   - `URL`: Enter the URL of the discussions server; for example: `http://discussions.example.com:8888/owc_discussions`.
   - `Admin User`: The user name of your discussions server administrator; for example, `admin`. This account is used by the Discussions and Announcement services to perform administrative operations on behalf of WebCenter Portal users.

     In WebCenter Portal: Spaces, this account is mostly used for managing space-related discussions and announcements. It is not necessary for this user to be a super admin. However, the user must have administrative privileges on the current root category for Spaces, that is, the category (on the discussions server) under which all space-related discussions and announcements are stored.

     Note: If your Framework application does not include space-related functionality, the administrator's user name is not required.
8. Optionally, enter a value for the other parameters:
   - `Connection Timeout`: Specify a suitable timeout for the connection. This is the length of time (in seconds) the application waits for a response from the discussions server before issuing a connection timeout message. The default is -1, which means that the service default is used. The service default is 10 seconds.

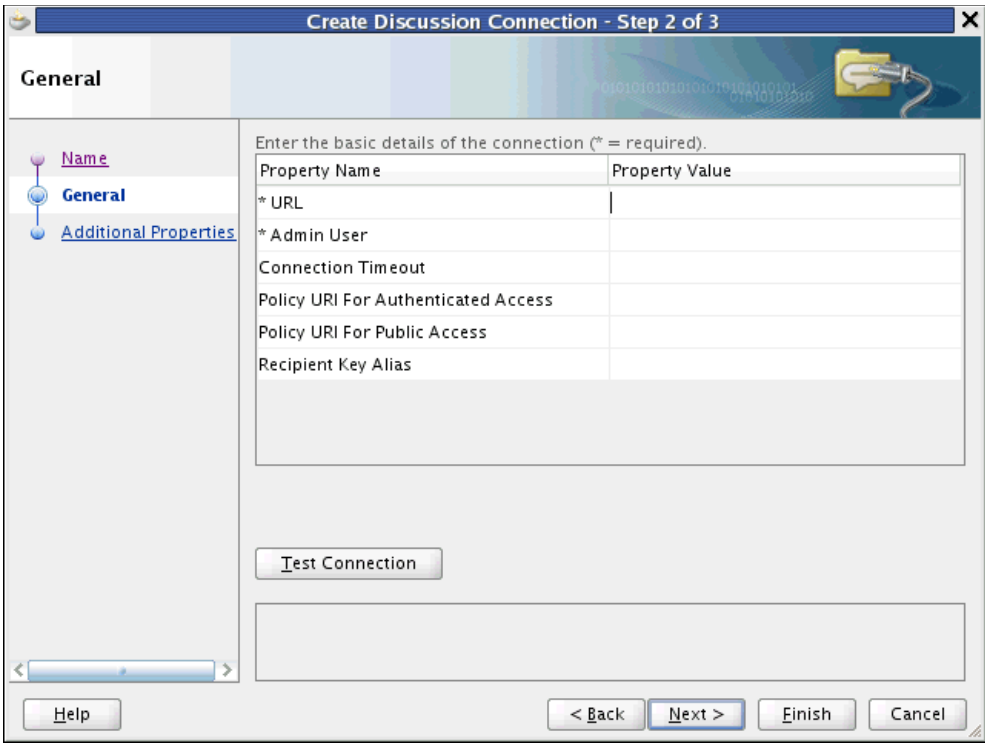
- `Policy URI for Authenticated Access`: Select the SAML token client policy this connection uses for authenticated access to the discussions server Web service. SAML (Security Assertion Markup Language) is an XML-based standard for passing security tokens defining authentication and authorization rights. An attesting entity (that has trust relationship with the receiver) vouches for the verification of the subject by method called sender-vouches. Options available are:
  - WSS 1.0 SAML Token Client Policy (oracle/wss10_saml_token_client_policy)
  - WSS 1.1 SAML Token With Message Protection Client Policy (oracle/wss11_saml_token_with_message_protection_client_policy)

  The client policy specified must be compatible with the service policy that is configured for the `OWCDiscussionsServiceAuthenticated` endpoint in the discussions server. Out-of-the-box, the default service policy is WSS 1.0 SAML Token Service Policy (oracle/wss10_saml_token_service_policy).
- `Policy URI for Public Access`: Select the client policy this connection uses to enforce message security and integrity for public access to the discussions server Web service. Options available are:
  - None - This is the default setting.
  - WSS 1.1 Message Protection Client Policy (oracle/wss11_with_message_protection_client_policy)

  The client policy specified must be compatible with the service policy that is configured for the `OWCDiscussionsServicePublic` endpoint in the discussions server. Out-of-the-box, a service policy is not configured for public access.
- `Recipient Key Alias`: Enter the recipient key alias to be used for message protected policies (applicable to the `OWCDiscussionsServicePublic` and `OWCDiscussionsServiceAuthenticated` endpoints). This is the alias to the certificate that contains the public key of the discussions server in the configured keystore.

  For more information, see the chapter "Configuring WS-Security for WebCenter Portal's Applications and Components" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Figure 33–5 Create Discussions Connection, Step 2



9. Click **Test Connection**, and if it is successful, then click **Next**.
10. On the **Create Discussion Connection - Step 3 of 3** page, you can configure additional parameters. For example:

    `application.root.category.id`: (WebCenter Portal: Spaces only) The application root category ID on the discussions server under which all discussion forums are stored. For example, if set to 3, then all forums are stored inside the category 3.

    Note: To encrypt property values, such as passwords, click **Add Secured Property**.

11. Click **Finish**. Your connection should now appear as a node under **Application Resources - Connections**.

33.2.2 Adding the Announcements Service at Design Time

This section explains a basic incorporation of the Announcement service.

33.2.2.1 Announcements Service Task Flows

Both the Announcements and the Announcements - Quick View task flows display current announcements. The Announcements task flow additionally offers tools for managing announcements within the task flow. With the Announcements - Quick View task flow, you must click the **Open Announcement Manager** icon to manage announcements, and administrators can configure this task flow to remove all announcement management functionality.

Table 33–1 Announcements Service Task Flows

Task Flow	Description
Announcements	This task flow displays a view that allows the user to see all current announcements and perform operations based on their privileges. For a Moderator, all command buttons are shown, but for a Reader, only the refresh and personalization options are shown. The personalization option lets users select the number of days to display announcements. The `parentId` parameter is the forum ID in the discussions server under which announcements are maintained. Each Framework application should create a forum on the discussions server.
Announcements - Quick View	This task flow displays a view that shows various categories of quick links to announcements. The `parentId` parameter is the forum ID in the discussions server under which announcements are maintained. Each Framework application should create a forum on the discussions server. The look and feel of this view changes with the optional parameter values given for rendering the task flow region. For information on how to add this task flow, see Section 33.3.1, "How to Add the Announcements - Quick View Task Flow."

33.2.2.2 How to Add Announcements to Your Page

The Announcements task flow provides a complete view of your announcements. To add the Announcements task flow to your Framework application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to implement security and create a new customizable page in your application.
2. Open the page on which you want to add the service.
3. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
4. Drag **Announcements** from the Resource Palette and drop it onto the page inside of the `af:form` `begin` and `end` tags.
5. When prompted, select **Region** as the way to create the task flow (and confirm with **Add Library**). This operation may take a moment to complete.
6. In the Edit Task Flow Binding dialog, enter a value for the `parentId` parameter.

   The discussions server could be shared with multiple Framework applications. Each Framework application should create a forum on the discussions server. Enter that forum ID here as the `parentId`; for example, `${2}`.
7. Click **OK**.
8. Save and run your page.

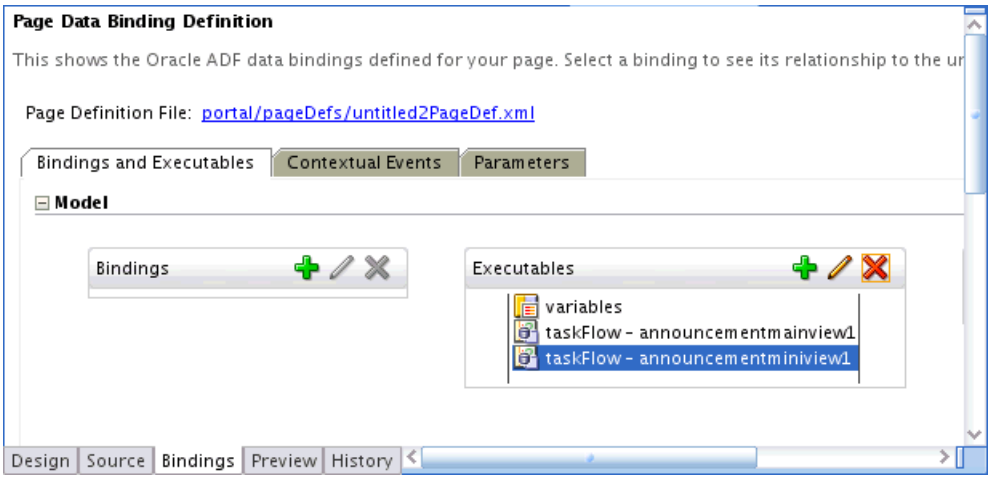
33.2.2.3 How to Modify Announcements Service Task Flow Parameters

The Announcements service task flows have required and optional task flow binding parameters.

You can adjust the parameter values when you drop the task flows onto a page or after you have placed a task flow on a page:

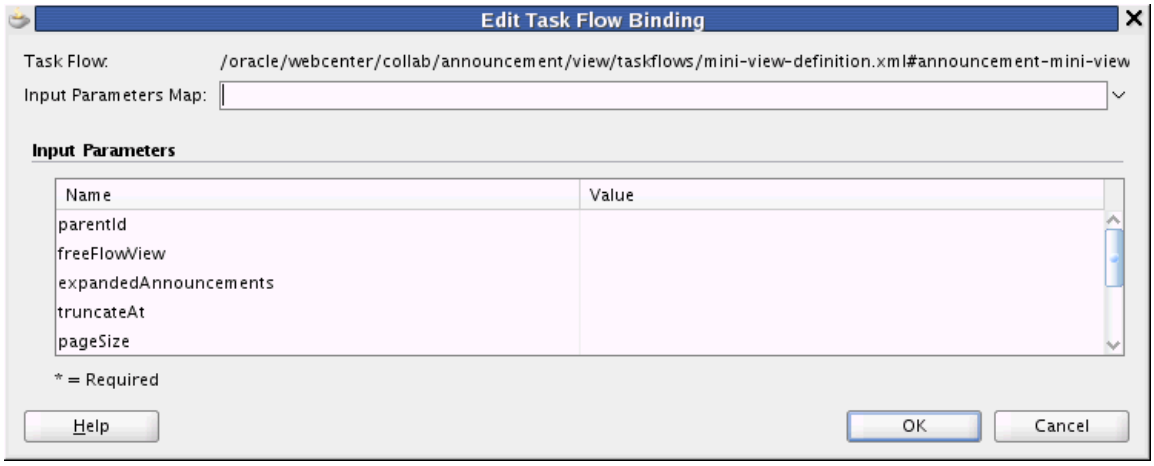
1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 33–6 shows an example of a Search task flow in the Executables section.

Figure 33–6 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog Figure 33–7, revise the binding parameter values as required.

Figure 33–7 Edit Task Flow Binding Dialog for Announcements - Quick View



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

Table 33–2 describes the properties that are unique to the Announcements service task flows.

Table 33–2 Announcements Service Task Flow Parameters

Property	Description	Task Flow
`parentId`	The forum ID in the discussions server under which announcement objects are maintained. Each Framework application should create a forum on the discussions server. Enter that forum ID here; for example, `${2}`. If this parameter is not specified, then announcements default to global announcements.	Announcements Announcements - Quick View
`freeFlowView`	A Boolean value representing whether to remove the announcement title (subject) and show the announcement body as is. The default value is `${false}`, which means that the announcement list displays the title and body in plain text and the `truncateAt` and `expandedAnnouncements` parameters control it. Enter `${true}` to remove the announcement title and ignore the values for `truncateAt` and `expandedAnnouncements`.	Announcements - Quick View
`expandedAnnouncements`	The number of announcements to display announcement details (that is, the body of the announcement). Users can click the title to display the full announcement content in rich text mode. The value you enter for `expandedAnnouncements` is ignored if `freeFlowView` is set to `true`.	Announcements - Quick View
`truncateAt`	For announcements that display announcement body, this value specifies how many characters to display for each announcement. Enter an Expression Language (EL) expression. For example, when the value is set to the EL expression `${50}`, following their titles, announcements display no more than 50 characters. Users can click announcement titles to display the full announcement. If no value is specified, then it displays 200 characters. If a non-valid positive integer value is specified, then it displays all characters in plain text. This parameter takes effect with `expandedAnnouncements`. The value you enter for `truncateAt` is ignored if `freeFlowView` is set to `true`.	Announcements - Quick View
`pageSize`	The number of announcements to show in a page on Extended Quick View.	Announcements - Quick View
`visibleAnnouncements`	The number of announcements to show in the Quick View.	Announcements - Quick View
`hideToolbar`	A Boolean value representing whether to display the task flow personalization feature.	Announcements - Quick View
`navigateToAnnouncementViewer`	A Boolean value representing whether to launch the announcement in a popup or navigate to the Announcement Manager. The default behavior is to launch in a popup. Enter `true` to navigate to the Announcement Manager.	Announcements - Quick View
`expandAllAnnouncements`	A Boolean value representing whether to display details for all announcements in extended quick view. Default value is `false`, in which case announcements display the announcement title only.	Announcements - Quick View

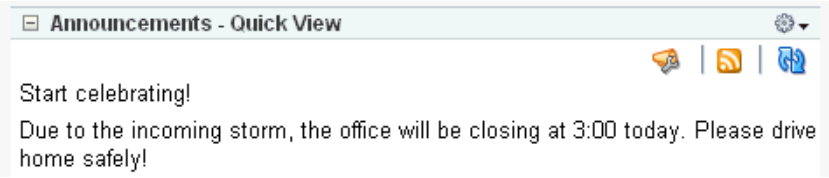
You can change the look and feel of the Announcements - Quick View with the parameter values. For example, Figure 33–12 shows the Announcements - Quick View at runtime with the `freeFlowView` parameter set to false (or empty).

Figure 33–8 Announcements - Quick View with Optional Parameters



Figure 33–13 shows the Announcements - Quick View at runtime with the `freeFlowView` parameter set to `${true}`.

Figure 33–9 Announcements - Quick View with freeFlowView Parameter



33.2.3 Setting Security for Announcements

By default, authenticated users of Framework applications can view and participate in announcements. Any user who has logged in to a WebCenter Portal application can view announcements, and users with sufficient permissions can create announcements. To create announcements, you must be a moderator or administrator on the back-end discussions server.

In an unsecured Framework application, identity propagation cannot happen. A user is a *guest* (or *anonymous*) user who can view only public categories and forums. In an unsecured application, the `parentId` (forum ID) from which announcements are fetched should be a public forum. If the forum is not public, then an error is reported. If the `parentId` parameter is not specified at all, then WebCenter Portal cannot fetch global announcements (that is, announcements not scoped to a forum): global announcements are not available for public users. Rather than relying to global announcements, each Framework application should reserve its own forum ID.

In an ADF-secured Framework application, identity propagation is enabled. Based on the identity, appropriate permissions are matched and corresponding actions are enabled. The user name that you use to log in to the application is used to log in to the discussions server. The recommended approach is to have the discussions server and the Framework application point to the same identity store. This way, your users can log in to the application one time and automatically connect to the discussions server.

> **Note:** The Announcements services requires that the identity store be LDAP-based; that is, not file-based with `jazn-data.xml`.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

33.3 Advanced Information for the Announcements Service

This section describes optional features available with this service. It includes the following subsections:

- Section 33.3.1, "How to Add the Announcements - Quick View Task Flow"
- Section 33.3.2, "Customizing Announcements Views"
- Section 33.3.3, "Obtaining Space RSS News Feed URL for the Announcements Service"

33.3.1 How to Add the Announcements - Quick View Task Flow

The Announcements - Quick View provides a snapshot (dashboard) view of the announcements (Figure 33–10).

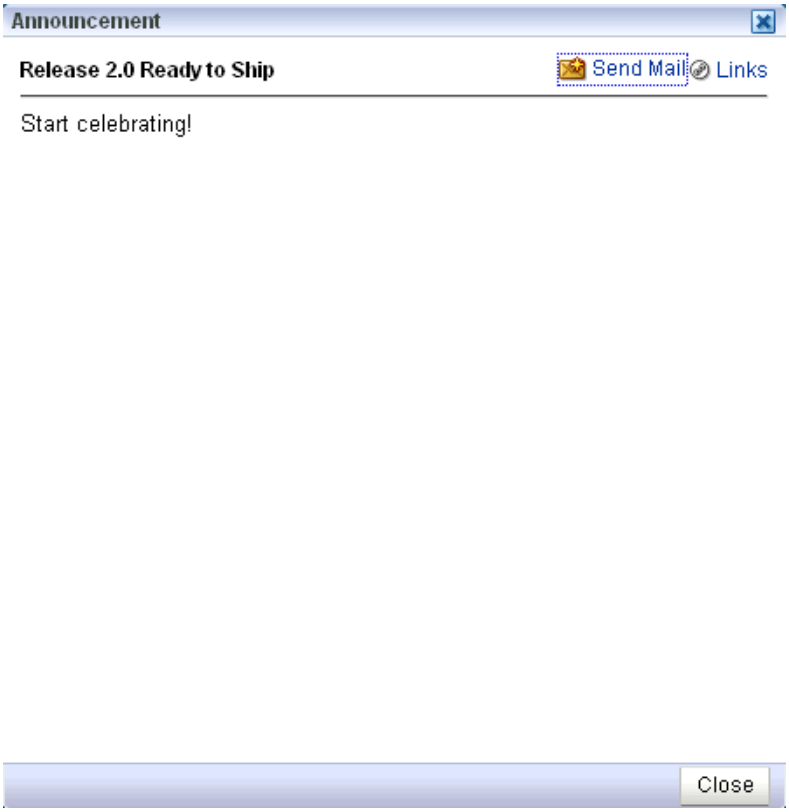
Figure 33–10 Announcements - Quick View



By default, announcements in the Announcements - Quick View task flow show announcement titles as links. But you can configure the task flow to display only announcement titles, titles with some amount of content, or only content.

Click an announcement to view it in a popup. From there, you can select the **Mail** icon to mail the announcement to anyone you choose or select the **Links** icon to link to this announcement (Figure 33–10).

Figure 33–11 Announcement Popup



The Announcements - Quick View task flow includes numerous parameters to customize your view. For example, you can remove the link to the Announcement Manager, to present announcements to end users where manage controls are not needed. The task flow lists 10 announcements by default, but you can change this number and how much of the announcement is displayed. The **More Announcements ...** link launches a popup containing the complete list of all announcements with pagination behavior. This is called the Extended Quick View (or Mini View).

To add the Announcements - Quick View to your Framework application, follow the same instructions that you did for the Announcements task flow in Section 33.2.2, "Adding the Announcements Service at Design Time," but drag and drop Announcements - Quick View onto the page.

Table 33–2 describes the Announcements - Quick View parameters. The look and feel of the Announcements - Quick View changes with the values provided for these parameters. For more information, see Section 33.2.2.3, "How to Modify Announcements Service Task Flow Parameters."

33.3.2 Customizing Announcements Views

You can change the look and feel of the Announcements - Quick View with the parameter values. For example, Figure 33–12 shows the Announcements - Quick View at runtime with the `freeFlowView` parameter set to false (or empty).

Figure 33–12 Announcements - Quick View with Optional Parameters

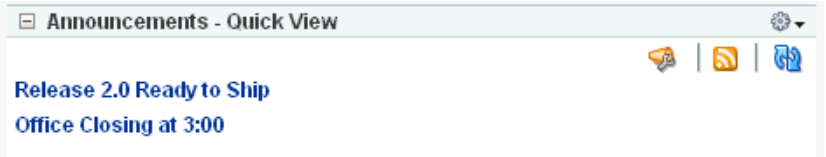
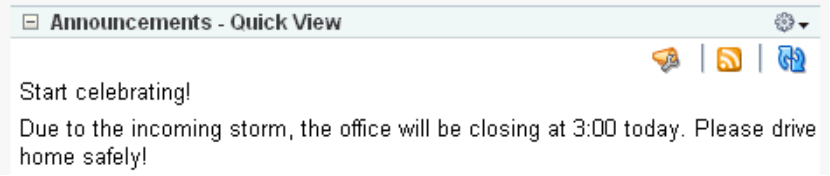


Figure 33–13 shows the Announcements - Quick View at runtime with the `freeFlowView` parameter set to `${true}`.

Figure 33–13 Announcements - Quick View with freeFlowView Parameter



33.3.3 Obtaining Space RSS News Feed URL for the Announcements Service

You can expose WebCenter Portal: Spaces functionality in a WebCenter Portal: Framework application. Your Framework application users can find out what is happening in a specific space through RSS news feeds.

Configure RSS news feeds for the Announcements service to enable users to view space announcements from within a Framework application. To obtain the space RSS news feed URL for the Announcements service, use either of the following APIs for Spaces:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the space (by name or GUID) and specify the service required (by service ID). The service ID for the Announcements service is `GroupSpaceWSClient.ANNOUNCEMENT_SERVICE_ID`.

For information about how to use these APIs, see Section 57.2.5.3.9, "Retrieving RSS Feed URLs for Space Services."

34

Integrating the Discussions Service

This chapter explains how to integrate the Discussions service in a WebCenter Portal: Framework application at design time. For more information about managing and including discussions, see:

- "Managing the Announcements and Discussions Services" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Discussions Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 34.1, "Introduction to Discussions"
- Section 34.2, "Basic Configuration for the Discussions Service"
- Section 34.3, "Advanced Information for the Discussions Service"

34.1 Introduction to Discussions

The Discussions service lets you expose discussion forums on your application pages, so users can create forums, post questions, and search for answers. For example, customers can share product reviews, or a customer service department can answer questions online. Discussion forums additionally provide the means to preserve and revisit discussions.

The back-end discussions server manages content in a hierarchy. At the top of the hierarchy are *categories*, below that are *forums*, and then *topics*. Within each topic are messages, and messages can be nested within messages.

Where categories are exposed in your application, authorized users can create multiple forums within a given scope and multiple topics under those forums. Where categories are not exposed, authorized users can create multiple topics under one forum within a given scope.

34.1.1 Understanding the Discussions Service

With the Discussions service, you can do the following (according to your permissions):

- Create a new discussion forum or a new topic.
- Navigate into a forum from a list of available forums.
- Edit, reply to, or delete an existing discussion forum, topic, or message.
- View the number of replies for a topic on the main forum view.

- Drill into a topic to read all replies by clicking the topic name.
- Add a "watch" to a topic or forum.
- View the following in a quick view:
  - Watched topics
  - Watched forums
  - Most popular (frequently viewed) topics
  - Most recent topics

> **Note:** In a secured application, Discussions permissions are allotted according to an individual user's assigned user role. A user can be a moderator, participant, or viewer. Some activities require moderator or participant roles. For more information, see Section 34.2.3, "Setting Security for Discussions."

The Discussions service is integrated with many WebCenter Portal services, such as Instant Messaging and Presence, RSS, and Search (to search within forums) services. Use the Mail service to archive mails into discussions as threads. Use the Links service to link to a discussion from any WebCenter Portal object.

34.1.2 Requirements for Discussions

The Discussions service requires a discussions server. Install and configure the discussions server that comes with Oracle Fusion Middleware.

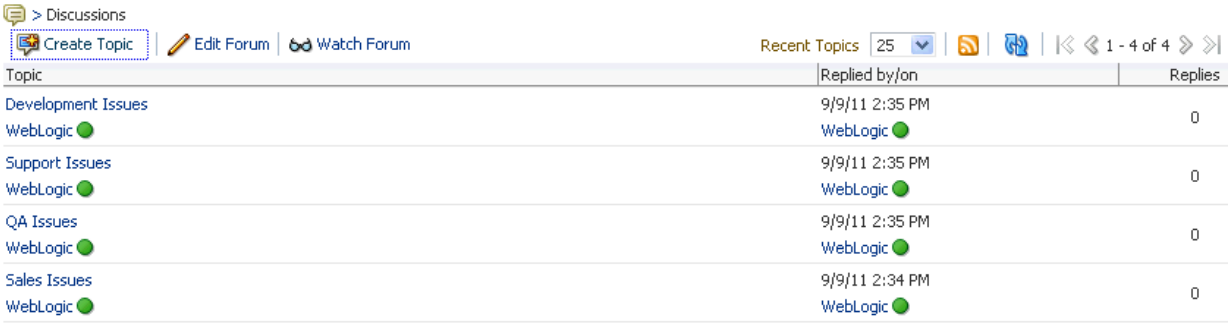
> **See Also:** *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*

34.1.3 What Happens at Runtime

To create forums in Framework applications, you must be a moderator or administrator on the back-end discussions server. For more information, see "Granting Administrator Role for Oracle WebCenter Portal's Discussions Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Figure 34–1 shows a discussion forum at runtime.

Figure 34–1 Discussion Forums Task Flow at Runtime



For more information about the service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

34.2 Basic Configuration for the Discussions Service

This section describes required steps for adding the Discussions service to your application. It includes the following subsections:

- Section 34.2.1, "Setting up Connections for Discussions"
- Section 34.2.2, "Adding the Discussions Service at Design Time"
- Section 34.2.3, "Setting Security for Discussions"

34.2.1 Setting up Connections for Discussions

You must create a connection to the discussions server in your WebCenter Portal: Framework application. You can register additional Discussion Forum connections, but only one connection is active at a time.

When you create a Discussion Forum connection or set a connection as active, both the Announcements and Discussions services use this same connection. If you have an existing connection, then you can skip this section and see Section 34.2.2, "Adding the Discussions Service at Design Time."

If you do not have an existing connection, then you must create a new Discussion Forum connection.

34.2.1.1 Discussions Connections

The Discussions service requires a Discussion Forum connection to the discussions server.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle Developer, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

34.2.1.2 How to Set Up Connections for Discussions

Follow these steps to set up the discussions connection.

1. In Oracle JDeveloper, open the application in which you plan to consume discussion forums.

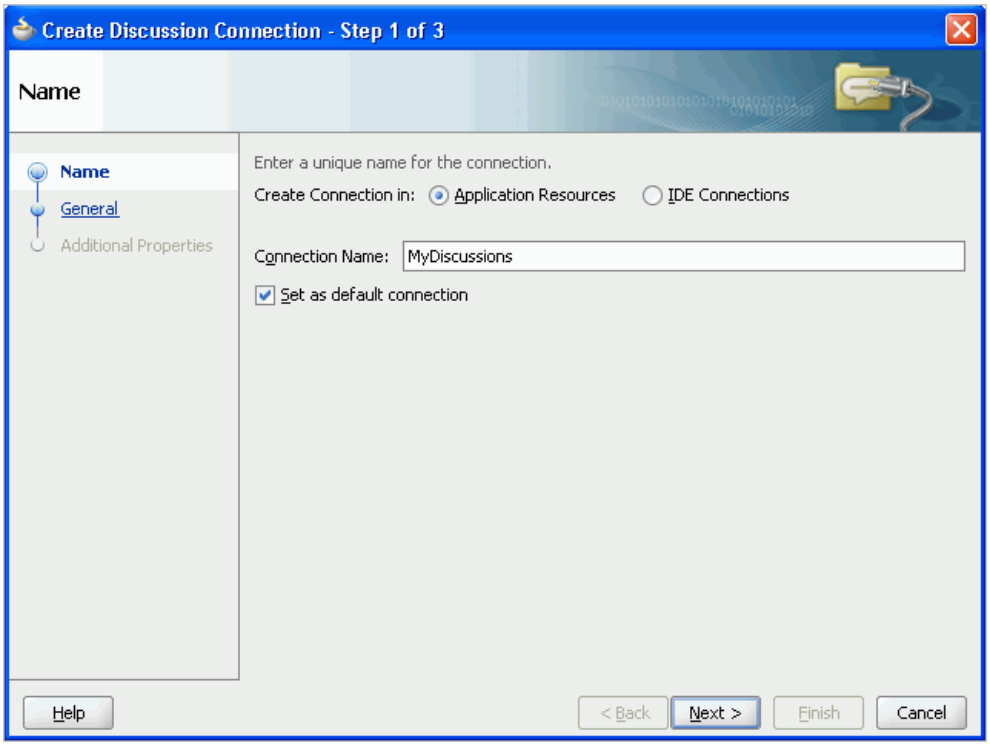
   > **Note:** If you created a Discussion Forum connection for the Announcements service, then that is used by default for the Discussions service. No extra configuration is required.

2. In the Application Navigator, under Application Resources, right-click **Connections**, then select the new connection **Discussion Forum** from the list.
3. On the **Name** page, select to create the connection in **Application Resources**. (A connection in Application Resources is available only for that application, while a connection in IDE Connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE Connections so you do not need to re-create it.)
4. For **Connection Name**, enter a unique name; for example, `MyDiscussions`.

5. Select the **Set the default connection** checkbox. You can have multiple connections, but only one can be active (default). If you have different discussions servers (for example, for each page in a space), then do not select the checkbox, but the service requires that one connection be marked as the default connection (Figure 34–2).

   Note: After you create a connection as the default connection, you cannot edit it so that it is *not* the default. To use a different default connection, you must create a new connection and mark that as the default connection.

Figure 34–2 Create Discussion Connections, Step 1



6. Click **Next**.

7. On the **General** page, enter values for the required parameters:

   - `URL`: Enter the URL of the discussions server hosting discussion forums and announcements. For example:
     `http://discuss-example.com:8888/owc_discussions`
   - `Admin User`: The user name of your discussions server administrator; for example, `admin`. This account is used by the Discussions and Announcement services to perform administrative operations on behalf of WebCenter Portal users.

     In WebCenter Portal: Spaces, this account is mostly used for managing space-related discussions and announcements. It is not necessary for this user to be a super admin. However, the user must have administrative privileges on the current root category for Spaces, that is, the category (on the discussions server) under which all space-related discussions and announcements are stored.

Note: If your Framework application does not include space-related functionality, then the administrator's user name is not required.

8. Optionally, enter a value for the other parameters:

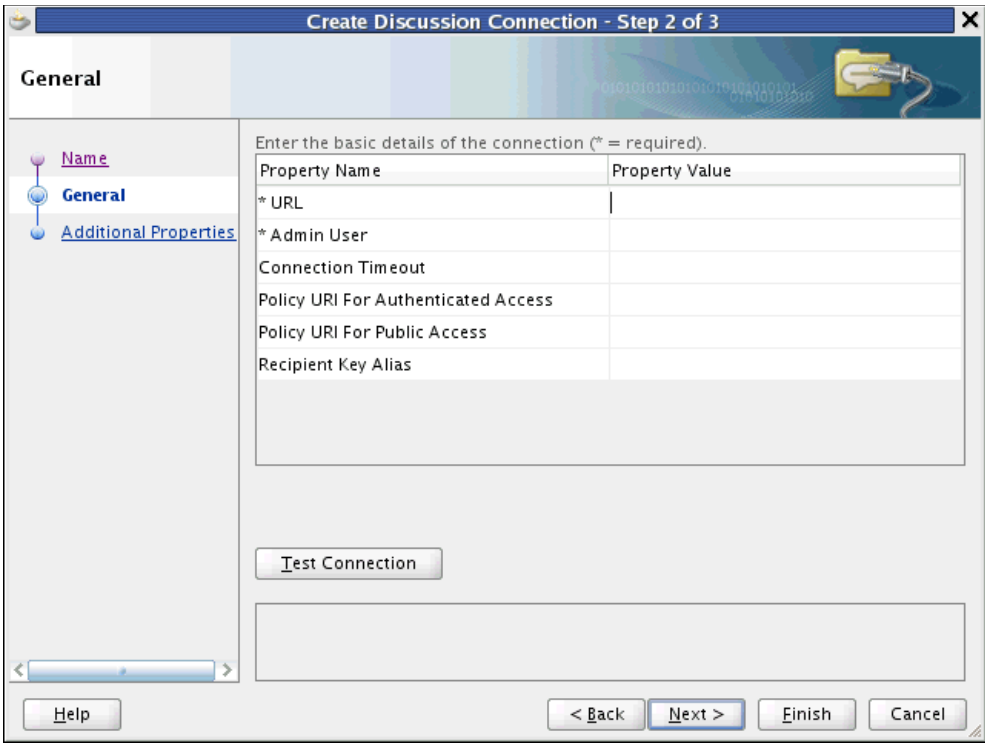
   - `Connection Timeout`: Specify a suitable timeout for the connection. This is the length of time (in seconds) the application waits for a response from the discussions server before issuing a connection timeout message. The default is -1, which means that the service default is used. The service default is 10 seconds.
   - `Policy URI for Authenticated Access`: Select the SAML token client policy this connection uses for authenticated access to the discussions server Web service. SAML (Security Assertion Markup Language) is an XML-based standard for passing security tokens defining authentication and authorization rights. An attesting entity (that has trust relationship with the receiver) vouches for the verification of the subject by method called sender-vouches. Options available are:
     - WSS 1.0 SAML Token Client Policy (oracle/wss10_saml_token_client_policy)
     - WSS 1.1 SAML Token With Message Protection Client Policy (oracle/wss11_saml_token_with_message_protection_client_policy)

     The client policy specified must be compatible with the service policy that is configured for the `OWCDiscussionsServiceAuthenticated` endpoint in the discussions server. Out-of-the-box, the default service policy is WSS 1.0 SAML Token Service Policy (oracle/wss10_saml_token_service_policy).
   - `Policy URI for Public Access`: Select the client policy this connection uses to enforce message security and integrity for public access to the discussions server Web service. Options available are:
     - None - This is the default setting.
     - WSS 1.1 Message Protection Client Policy (oracle/wss11_with_message_protection_client_policy)

     The client policy specified must be compatible with the service policy that is configured for the `OWCDiscussionsServicePublic` endpoint in the discussions server. Out-of-the-box, a service policy is not configured for public access.
   - `Recipient Key Alias`: Enter the recipient key alias to be used for message protected policies (applicable to the `OWCDiscussionsServicePublic` and `OWCDiscussionsServiceAuthenticated` endpoints). This is the alias to the certificate that contains the public key of the discussions server in the configured keystore.

     For more information, see the chapter "Configuring WS-Security for WebCenter Portal's Applications and Components" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Figure 34–3 Create Discussions Connection, Step 2



9. Click **Test Connection**, and if it is successful, then click **Next**.
10. On the **Create Discussion Connection - Step 3 of 3** page, you can configure additional parameters.

    `application.root.category.id`: (WebCenter Portal: Spaces only) The application root category ID on the discussions server under which all discussion forums are stored. For example, if set to 3, then all forums are stored inside the category 3.

    Note: To encrypt property values, such as passwords, click **Add Secured Property**.

11. Click **Finish**. Your connection should now appear as a node under **Application Resources - Connections**.

34.2.2 Adding the Discussions Service at Design Time

This section explains a basic incorporation of the Discussions service.

34.2.2.1 Discussions Task Flows

The Discussions service provides several task flows (Table 34–1) to enable you to include the service in a form that best suits your needs.

Table 34–1 Discussions Service Task Flows

Task Flow	Description
Discussion Forums	This task flow displays the Discussion Forums view, which allows the user to see all the discussions and their respective replies. It also allows users to perform various operations based on their privileges. A Moderator can perform create, read, update, and delete operations on all objects. A Participant can create a topic, edit a topic that has been created by him, and reply to a topic. A Viewer can only view objects. All watched forums and topics are accessible from this task flow. The Watched Forums and Watched Topics task flows provide more focussed views of watched forums or watched topics. The parameters alter the way the view appears. For more information, see Section 34.2.2.3, "How to Modify the Discussions Service Task Flow Parameters."
Discussions - Popular Topics	This task flow provides a view that allows users to see the most frequently viewed topics in the application under a given category ID or forum ID. For more information, see Section 34.3.1, "Adding the Discussions - Popular Topics Task Flow."
Discussions - Recent Topics	This task flow displays a view that allows users to see all the recent topics in the application given a category ID or forum ID. For more information, see Section 34.3.2, "Adding the Discussions - Recent Topics Task Flow."
Discussions - Watched Forums	This task flow displays a view that allows users to see all of their watched forums in the application under a given category ID. For more information, see Section 34.3.3, "Adding the Discussions - Watched Forums Task Flow."
Discussions - Watched Topics	This task flow displays a view that allows users to see all their watched topics in the application under a given category ID or forum ID. For more information, see Section 34.3.4, "Adding the Discussions - Watched Topics Task Flow."
Discussions - Quick View	This task flow displays a combined view of the Popular Topics, Recent Topics, Watched Topics, and Watched Forums task flows. Instead of adding four separate task flows, this single task flow presents all four views with a dropdown list so that end users can personalize it For more information, see Section 34.3.5, "Adding the Discussions - Quick View Task Flow."

34.2.2.2 How to Add Discussions to a Page

The Discussion Forums task flow provides a complete view of your discussions. To add the Discussion Forums task flow to your Framework application, follow these steps.

1. Follow the steps described in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security and create a new customizable page in your application.
2. Open the page on which you want to add the Discussions service.
3. In the Resource Palette, expand **My Catalogs**, **WebCenter - Framework Services Catalog**, and **Task Flows**.

4. Drag **Discussions** from the Resource Palette and drop it onto the page.
5. When prompted, select **Region** as the way to create the task flow.
6. You may be prompted to add the discussions library to your project. If so, then click **Add Library**. This operation may take a moment to complete.
7. In the Edit Task Flow Binding dialog box, enter values for the parameters, and click **OK**. Table 34–2 describes the parameters.
8. Save and run your page.

Notes: If you created a connection in IDE and not in the application, then the connection must be added to the application. For example, in the Resource Palette under IDE Connections, right-click the connection and select **Add to Application**.

All instances of the Discussion Forums task flow in an application run against the same discussions server: it is unnecessary to add multiple Discussion Forums task flow instances. This is true for all service task flows that require connections to back-end servers, such as task flows from the Announcements or Mail services.

The Discussion Forums main view contains some features that require other services in your application.

- For the links in the main view to work, you must have configured the Links service. For more information, see Chapter 43, "Integrating the Links Service."
- For users' presence indicators to work, you must have configured the WebCenter Portal Presence service. For more information, see Chapter 35, "Integrating the Instant Messaging and Presence Service."
- Oracle recommends that the discussions server and WebCenter Portal point to the same identity store. If the Framework application and the discussions page are configured to be secure, then upon logging in to the application, then the Discussions service respects those credentials in the discussions server.

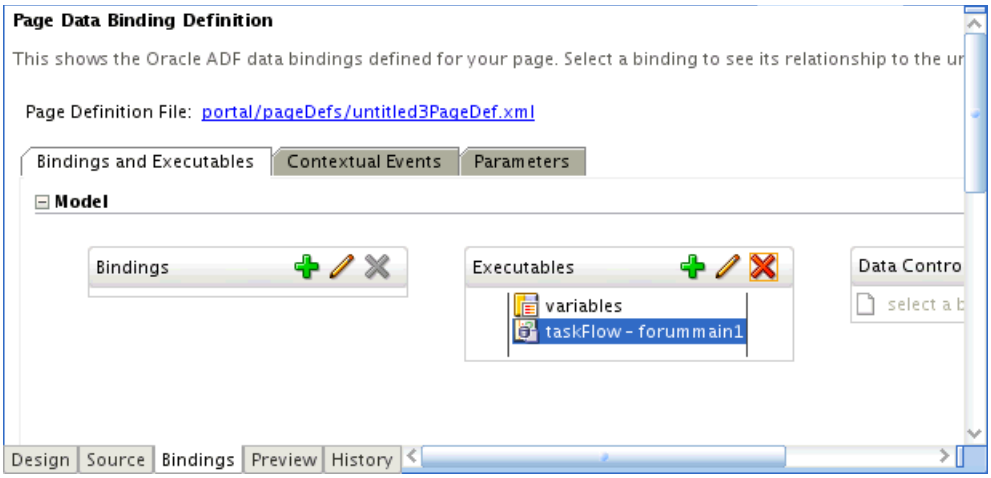
34.2.2.3 How to Modify the Discussions Service Task Flow Parameters

The Discussions service task flows have optional task flow binding parameters.

You can adjust the parameter values when you drop the task flows onto a page or after you have placed a task flow on a page:

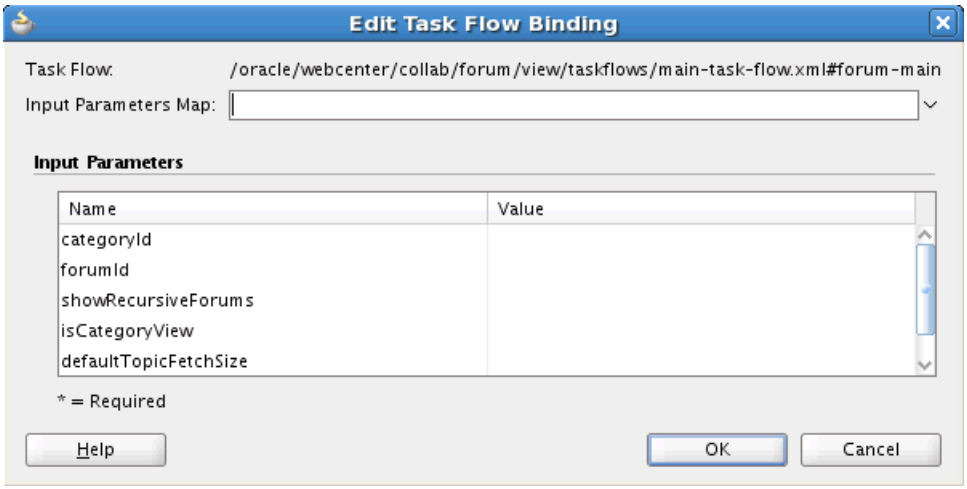
1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 34–4 shows an example of a Search task flow in the Executables section.

Figure 34–4 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog Figure 34–5, revise the binding parameter values as required.

Figure 34–5 Edit Task Flow Binding Dialog for Discussion Forums Task Flow



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

Table 34–2 describes the properties that are unique to the Discussions service task flows.

Table 34–2 Discussions Service Task Flow Parameters

Property	Description	Task Flow
`categoryId`	This optional parameter is an identifier for an existing category in Oracle WebCenter Portal's Discussion Server to which the view should be scoped. When no value is supplied, it defaults to the appropriate root category of the discussions server. (This root category ID can be overridden by supplying an additional property named `application.root.category.id` in the connection.) For testing purposes, you may want to create a category through the discussions server administrator interface and then reference that category identifier here.	■ Discussion Forums ■ Discussions - Quick View ■ Popular Topics ■ Recent Topics ■ Watched Forums ■ Watched Topics
`forumId`	This parameter is an identifier for an existing forum in your discussions server for which popular topics should be fetched. If both `categoryId` and `forumId` are given, then only `categoryId` is honored.	■ Discussion Forums ■ Popular Topics ■ Recent Topics ■ Watched Topics
`disableToolbar`	This parameter indicates whether the toolbar, including the Refresh icon, should be displayed. It can be `${true}`, `${false}`, or undefined. If you leave this parameter undefined, then the default behavior (false) is to show the toolbar.	■ Popular Topics ■ Recent Topics ■ Watched Forums ■ Watched Topics
`showRecursiveForums`	This parameter determines if you show forums either in a category only or in subcategories. True means all forums under a given category/subcategory are shown; false means only the category's direct child forums are shown. The default value is false. Note: A value of true can impact performance.	Discussion Forums
`isCategoryView`	A means of showing the forums grouped under the Category ID category or the topics specified under the Forum ID forum. True means you want the task flow to display the forums classified under `categoryId`; false, the default value, means you want the task flow to display the topics associated with the specified `forumId`. This parameter value works in combination with other parameters.	Discussion Forums
`defaultTopicFetchSize`	Sets the number of visible topics	Discussion Forums
`doNotAllowSelectingPageSize`	If set to true, users are not allowed to change the number of visible topics.	Discussion Forums
`visibleTopicsFetchSize`	Sets the number of visible watched topics.	■ Recent Topics ■ Watched Topics
`visibleForumsFetchSize`	Sets the number of visible watched forums.	Watched Forums
`doNotShowMoreLink`	If set to true, the **More** link is not visible.	■ Discussions - Quick View ■ Recent Topics ■ Watched Forums ■ Watched Topics

You can customize the look and feel of **Discussion Forums** views by changing parameter values. The following combinations are possible:

> **Note:** A *bidirectional link* connects back and forth. For example, when you create a link from a discussion topic to a document, a link from the document back to the topic also is created. Similarly, when you delete the link from the discussion topic to a document, the link from the document back to the topic automatically is deleted.

- `categoryId`: This displays the forums list view if there are multiple forums. If there is only one forum, then it drills into the forum and lists all topics with a bidirectional link enabled.
- `categoryId` and `forumId`: This displays the topics list view with a bidirectional link enabled.
- `isCategoryView` is set to `true`: This displays the topics list view with a bidirectional link enabled.
- `categoryId` and `forumId` and `isCategoryView` is set to `false`: This displays the topics list view, but a bidirectional link is not enabled.
- `categoryId` and `isCategoryView` is set to `true`: This displays the forums list view if there are multiple forums. If there is only one forum, then it displays that forum with a bidirectional link enabled.
- `forumId` and `isCategoryView` is set to `false`: This displays the topics list view, but a bidirectional link is not enabled.
- `categoryId` and `isCategoryView` is set to `false`: This is similar to when `categoryId` alone is given.
- `forumId` and `isCategoryView` is set to `true`: This is similar to when `forumId` and `isCategoryView` is set to false.
- `isCategoryView` = `true` or `false`: This is ignored. It goes with the default scope (that is, all forums listed under the root category) in a Framework application.
- `forumId`: This is similar to `forumId` and `isCategoryView` is set to false.

34.2.3 Setting Security for Discussions

By default, authenticated users of WebCenter Portal applications can view and participate in discussion forums. Any user who has logged in to a WebCenter Portal application can view discussions, and users with sufficient permissions can create forums and topics. To create forums, you must be a moderator or administrator on the back-end discussions server that provides the Discussions service.

In an unsecured Framework application, the user identity is not propagated to the discussions server; consequently, users are identified as GUEST (that is, as anonymous users) and can view only public categories and forums.

> **Note:** *Categories* and *forums* are server-side classifications that move from *category* to *forum* to *topic*. That is, a category can head a collection of forums, just as a forum can head a collection of topics.

In a secured Framework application, discussions permissions are allotted according to an individual user's assigned user role. For example, a user can be a moderator, participant, or viewer. A forum moderator can edit and delete any topics and messages within a forum. A forum participant can create topics and edit his or her own topics. A forum viewer can view topics and messages.

In an unsecured Framework application, identity propagation cannot happen. A user is a *guest* (or *anonymous*) user who can view only public categories and forums.

In an ADF-secured Framework application, identity propagation is enabled. Based on the identity, appropriate permissions are matched and corresponding actions are enabled. The user name that you use to log in to the application is used to log in to the discussions server. The recommended approach is to have the discussions server and the application point to the same identity store. When you run the page, you are presented with a login page for user credentials. Enter the credentials with the required privileges in the discussions server.

Note: The Discussions services requires that the identity store be LDAP-based; that is, not file-based with `jazn-data.xml`.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

34.3 Advanced Information for the Discussions Service

This section describes optional features available with this service. It includes the following subsections:

- Section 34.3.1, "Adding the Discussions - Popular Topics Task Flow"
- Section 34.3.2, "Adding the Discussions - Recent Topics Task Flow"
- Section 34.3.3, "Adding the Discussions - Watched Forums Task Flow"
- Section 34.3.4, "Adding the Discussions - Watched Topics Task Flow"
- Section 34.3.5, "Adding the Discussions - Quick View Task Flow"
- Section 34.3.6, "Obtaining Space RSS News Feed URL for the Discussions Service"
- Section 34.3.7, "Using Custom Discussions APIs"
- Section 34.3.8, "Using the Discussions Service REST APIs"
- Section 34.3.9, "Troubleshooting the Discussions Service"

34.3.1 Adding the Discussions - Popular Topics Task Flow

The Discussions - Popular Topics task flow provides a view that allows users to see the most frequently viewed discussion topics under a given category ID or forum ID.

Note: Popular topics are based on topic replies. The administrator is not able to determine popularity.

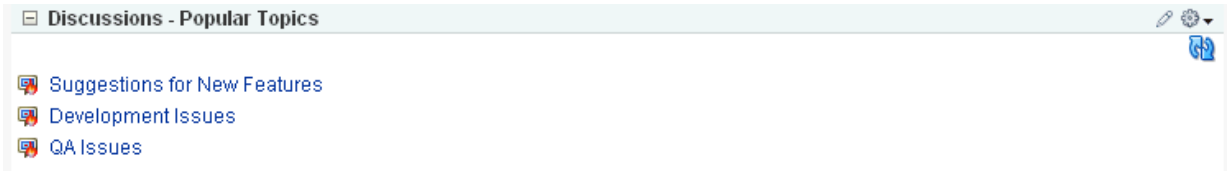
To add the Discussions - Popular Topics task flow to your Framework application, follow the same instructions that you did for the Discussion Forums task flow in

Section 34.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Popular Topics onto the page.

Table 34–2 describes the available parameters for this task flow.

Figure 34–6 shows how the Discussions - Popular Topics task flow looks at runtime.

Figure 34–6 Discussions - Popular Topics View at Runtime



34.3.2 Adding the Discussions - Recent Topics Task Flow

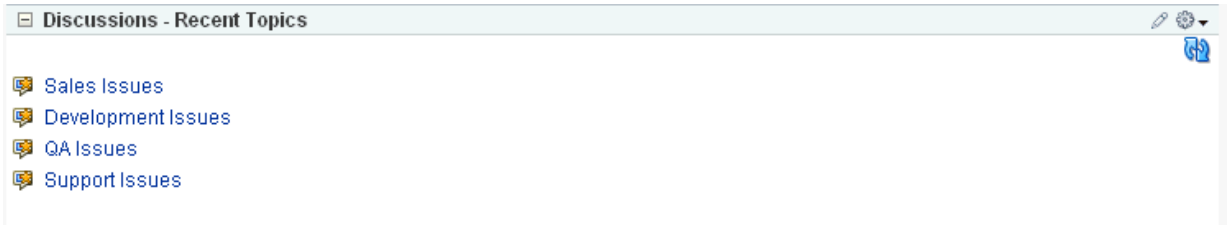
The Discussions - Recent Topics task flow displays a view that allows users to see all the recent topics (that is, topics posted within the past two days) given a category ID or forum ID.

To add the Discussions - Recent Topics task flow to your Framework application, follow the same instructions that you did for the Discussion Forums task flow in Section 34.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Recent Topics onto the page.

Table 34–2 describes the available parameters for this task flow.

Figure 34–7 shows how the Discussions - Recent Topics task flow looks at runtime. To view a topic post, simply click the relevant topic.

Figure 34–7 Discussions - Recent Topics View at Runtime



34.3.3 Adding the Discussions - Watched Forums Task Flow

The Discussions - Watched Forums task flow displays a view that allows users to see all of their watched forums under a given category ID.

You can watch discussion forums and topics to keep a close eye on the information most current and relevant to your efforts. The forums and topics you select to watch are personal, in that your selections appear on your view of watch lists. No other user is affected by the forums and topics you choose to watch.

When a user places a watch on a forum or a topic, whenever users add to that forum or topic, in addition to it appearing in the user's watched forums or topics list, the user receives an mail notification.

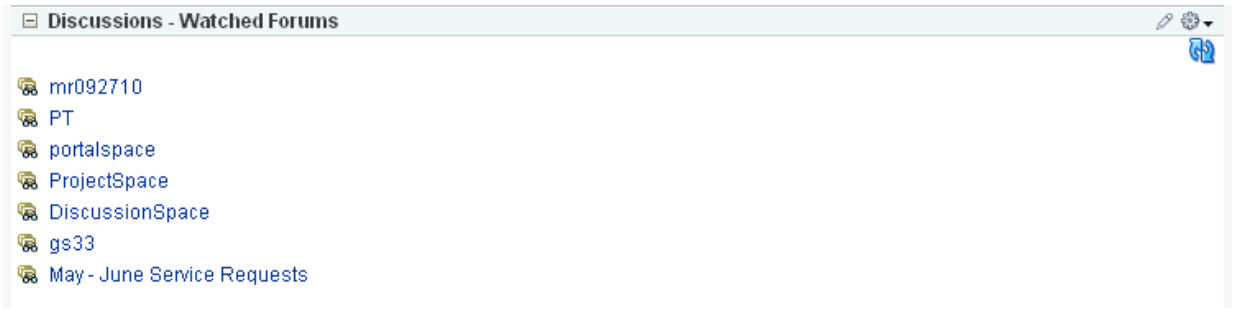
All watched forums and topics are accessible from the Forums task flow. The Watched Forums and Watched Topics task flows provide more focussed views of watched forums or watched topics.

To add the Discussions - Watched Forums task flow to your Framework application, follow the same instructions that you did for the Discussion Forums task flow in Section 34.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Watched Forums onto the page.

Table 34–2 describes the available parameters for this task flow.

Figure 34–8 shows how the Discussions - Watched Forums task flow looks at runtime.

Figure 34–8 Discussions - Watched Forums View at Runtime



For information on how to watch a forum or topic at runtime, see "Working with the Discussions Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces.*

34.3.4 Adding the Discussions - Watched Topics Task Flow

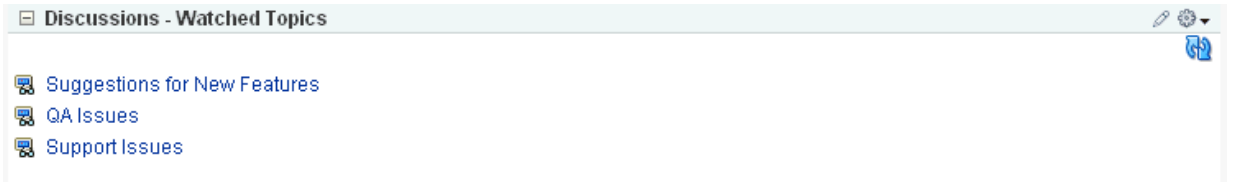
Similar to the Discussions - Watched Forums task flow, the Discussions - Watched Topics task flow displays a view that allows users to see all their watched topics under a given category ID or forum ID.

To add the Discussions - Watched Topics task flow to your Framework application, follow the same instructions that you did for the Discussion Forums task flow in Section 34.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Watched Topics onto the page.

Table 34–2 describes the available parameters for this task flow.

Figure 34–9 shows how the Discussions - Watched Topics task flow looks at runtime.

Figure 34–9 Discussions - Watched Topics View at Runtime



For information on how to watch a forum or topic at runtime, see "Working with the Discussions Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces.*

34.3.5 Adding the Discussions - Quick View Task Flow

The Discussions - Quick View task flow displays a combined view of the Popular Topics, Recent Topics, Watched Topics, and Watched Forums task flows. Instead of

adding four separate task flows, this single task flow presents all four views with a dropdown list so that end users can personalize it.

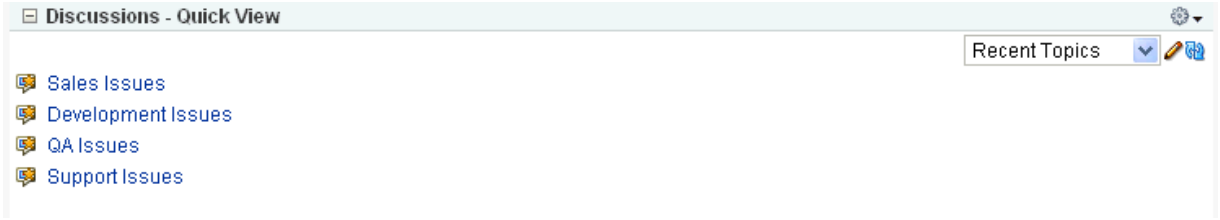
By default, Watched Topics are displayed. This task flow takes only one parameter: `categoryId`.

To add the Discussions - Quick View task flow to your WebCenter Portal: Framework application, follow the same instructions that you did for the Discussion Forums task flow in Section 34.2.2.2, "How to Add Discussions to a Page," but drag and drop Discussions - Quick View onto the page.

Table 34–2 describes the available parameters for this task flow.

Figure 34–10 shows how the Discussions - Quick View task flow looks at runtime.

Figure 34–10 Discussions - Quick View at Runtime



34.3.6 Obtaining Space RSS News Feed URL for the Discussions Service

You can expose WebCenter Portal: Spaces functionality in a Framework application. Your Framework application users can find out what is happening in a specific space through RSS news feeds.

Configure RSS news feeds for the Discussions service to enable users to view space discussions from within a Framework application. To obtain the space RSS news feed URL for the Discussions service, use either of the following Spaces APIs:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the space (by name or GUID) and specify the service required (by service ID). The service ID for the Discussions service is `GroupSpaceWSClient.DISCUSSION_SERVICE_ID`.

For information about how to use these APIs, see Section 57.2.5.3.9, "Retrieving RSS Feed URLs for Space Services."

34.3.7 Using Custom Discussions APIs

The back-end Oracle WebCenter Portal's Discussion Server includes APIs to further customize your application. To learn more about them, see the Jive Forums documentation on the Oracle Fusion Middleware documentation library (in the WebCenter Portal product area).

34.3.8 Using the Discussions Service REST APIs

WebCenter Portal provides REST APIs to support the Discussions service. Use the Discussions service REST APIs to post, read, update, and delete discussion forums, topics, and messages.

This section describes the REST APIs associated with the Discussions service. It includes the following subsections:

- Section 34.3.8.1, "Discussions Entry Point"
- Section 34.3.8.2, "Discussions Resource Type Taxonomy"
- Section 34.3.8.3, "Security Considerations"
- Section 34.3.8.4, "Discussions Resource Types"

For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

34.3.8.1 Discussions Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. To find the entry point for the Discussions service, find the link element with a `resourceType` of:

```
urn:oracle:webcenter:discussions:forums
```

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Discussions service.

For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

34.3.8.2 Discussions Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 54.5.2.1, "Resource Type."

The taxonomy for the Discussions service is:

```
urn:oracle:webcenter:discussions:forums
   urn:oracle:webcenter:discussions:forum
   urn:oracle:webcenter:discussions:forum:topics
      urn:oracle:webcenter:discussions:forum:topic
      urn:oracle:webcenter:discussions:forum:topic:messages
         urn:oracle:webcenter:discussions:forum:topic:message
```

Beyond the service entry points, URL templates allow clients to pass query parameters to customize their requests and control the form of returned data.

Collection resources in the discussions resources support pagination (`startIndex` and `itemsPerPage`). Other query parameters are not supported (that is, `search` and `projection`).

34.3.8.3 Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

34.3.8.4 Discussions Resource Types

This section provides the information about each resource type. It includes the following subsections:

- Section 34.3.8.4.1, "urn:oracle:webcenter:discussions:forums"
- Section 34.3.8.4.2, "urn:oracle:webcenter:discussions:forum"
- Section 34.3.8.4.3, "urn:oracle:webcenter:discussions:forum:topics"
- Section 34.3.8.4.4, "urn:oracle:webcenter:discussions:forum:topic"
- Section 34.3.8.4.5, "urn:oracle:webcenter:discussions:forum:topic:messages"
- Section 34.3.8.4.6, "urn:oracle:webcenter:discussions:forum:topic:message"

34.3.8.4.1 urn:oracle:webcenter:discussions:forums Use this resource type to identify the URI to use to read (`GET`) and write (`POST`) discussion forums. The response from a `GET` operation includes each forum in this collection of forums, and each forum includes links used to operate on that forum. The response from a `POST` operation includes the forum that was created in this collection of forums and a link to operate on that forum.

Navigation Paths to forums

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
```

Supported Methods for forums

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters**: `startIndex`, `itemsPerPage` (pagination)
  - **response - body:** forums
- `POST`
  - **request - body:** forum
  - **response - body:** forum

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to From forums

Table 34–3 lists the resource types that the client can link to from this resource.

Table 34–3 Related Resource Types for forums

rel	resourceType
self	`urn:oracle:webcenter:discussions:forums`
	`urn:oracle:webcenter:discussions:forum`

34.3.8.4.2 urn:oracle:webcenter:discussions:forum Use this resource type to identify the URI to use to read (`GET`), update (`PUT`), and delete (`DELETE`) a specific discussion forum. The response from a `GET` operation includes the specific forum identified by the URI. The response from a `PUT` operation includes the modified version of the forum identified by the URI. The response from a `DELETE` operation is a 204.

Navigation Paths to forum

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums
      forum

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            forum

resourceindex
   activities
      forum
```

Supported Methods for forum

The following methods are supported by this resource:

- `GET`
  - **request - body:** none
  - **response - body:** forum
- `PUT`
  - **request - body:** forum
  - **response - body:** forum
- `DELETE`
  - **request - body:** none
  - **response - body:** none

Writable Elements for forum

Table 34–4 lists the writable elements for this resource.

Table 34–4 Writable Elements for forum

Element	Type	Required	Constraints	Description
`name`	String	Yes	1 or more characters	Name of the forum
`displayName`	String	No	1 or more characters	Name used in presentation
`description`	String	No	1 or more characters	Description of the forum

Read-only Elements for forum

Table 34–5 lists the read-only elements for this resource.

Table 34–5 Read-only Elements for forum

Element	Type	Description
`id`	Integer	ID of the forum

Table 34–5 (Cont.) Read-only Elements for forum

Element	Type	Description
parentId	Integer	ID of the parent category
createdBy	String	ID of the user that created the forum
author	personReference	User information about the user that created the forum, including GUID, ID, display name, and a link to the profile icon (same user as createdBy)
createdOn	Date	Date on which the forum was created
updatedBy	String	ID of the user that performed the last modification
modifiedBy	personReference	User information about the user that performed the last modification, including GUID, ID, display name, and a link to the profile icon (same user as updatedBy)
updatedOn	Date	Date on which the forum was last modified
webUrl	String	URL for direct access to the discussions server
topicCount	Integer	Number of topics
messageCount	Integer	Number of messages
locked	Boolean	True if this forum is locked
favorite	Boolean	True if this forum is marked as a favorite

Resource Types Linked to From forum

Table 34–6 lists the resource types that the client can link to from this resource.

Table 34–6 Related Resource Types for forum

rel	resourceType
self	urn:oracle:webcenter:discussions:forum
	urn:oracle:webcenter:discussions:forum:topics

34.3.8.4.3 urn:oracle:webcenter:discussions:forum:topics Use this resource type to identify the URI to use to read (GET) and write (POST) discussion topics. The response from a GET operation includes each topic in this collection of topics, and each topic includes links used to operate on that topic. The response from a POST operation includes the topic that was created in this collection of topics and a link to operate on that topic.

Navigation Paths to topics

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums
      topics

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            topics
```

Supported Methods for topics

The following methods are supported by this resource:

- GET
  - **request - body:** none, **Parameters:** `startIndex`, `itemsPerPage` (pagination)
  - **response - body:** topics
- POST
  - **request - body:** topic
  - **response - body:** topic

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to From topics

Table 34–7 lists the resource types that the client can link to from this resource.

Table 34–7 Related Resource Types for topics

rel	resourceType
self	`urn:oracle:webcenter:discussions:forum:topics`
	`urn:oracle:webcenter:discussions:forum:topic`

34.3.8.4.4 urn:oracle:webcenter:discussions:forum:topic Use this resource type to identify the URI to use to read (`GET`), update (`PUT`), and delete (`DELETE`) a specific discussion topic. The response from a `GET` operation includes the specific topic identified by the URI. The response from a `PUT` operation includes the modified version of the topic identified by the URI. The response from a `DELETE` operation is a 204.

Navigation Paths to topic

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums
      topics
         topic

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            topics
               topic

resourceindex
   activities
      topicT
```

Supported Methods for topic

The following methods are supported by this resource type:

- GET
  - **Request—Body:** none
  - **Response—Body:** topic

- PUT
  - **Request—Body:** topic
  - **Response—Body:** topic
- DELETE
  - **Request—Body:** none
  - **Response—Body:** none

Writable Elements for topic

Table 34–8 lists the writable elements for this resource type.

Table 34–8 Writable Elements for topic

Element	Type	Required	Constraints	Description
subject	String	Yes	1 or more characters	Subject of the topic
body	String	No	0 or more characters	Contents of the topic

Read-only Elements for topic

Table 34–9 lists the read-only elements for this resource type.

Table 34–9 Read-only Elements for topic

Element	Type	Description
id	Integer	Identifier for the topic
parentId	Integer	Identifier for the parent message
forumId	Integer	Identifier for the forum under which this topic is posted
topicId	Integer	Identifier for the topic under which this topic is posted
createdBy	String	ID of the user who created the topic
author	personReference	User information about the user who created the topic, including GUID, ID, display name, and a link to the profile icon (same user as createdBy)
createdOn	Date	Date on which the topic was created
updatedBy	String	User who lasted updated the topic
updatedOn	Date	Date on which the topic was last updated
webUrl	String	URL for direct access to the discussions server
depth	Integer	Depth in the hierarchy of messages
messageCount	Integer	Number of child messages under this topic
numberOfReplies	Integer	Number of replies to this topic
favorite	Boolean	Whether the topic is marked as a favorite for the user
locked	Boolean	Whether the topic is locked
hidden	Boolean	Whether the topic is hidden

Table 34–9 (Cont.) Read-only Elements for topic

Element	Type	Description
`hasAttachment`	Boolean	Whether the topic has attachments

Resource Types Linked to From topic

Table 34–10 lists the resource types that the client can link to from this resource.

Table 34–10 Related Resource Types for topic

rel	resourceType
self	`urn:oracle:webcenter:discussions:forum:topic`
	`urn:oracle:webcenter:discussions:forum:topic:messages`

34.3.8.4.5 urn:oracle:webcenter:discussions:forum:topic:messages Use this resource type to identify the URI to use to read (`GET`) and write (`POST`) discussion topic messages. The response from a `GET` operation includes each message in this collection of messages, and each message includes links used to operate on that message. The response from a `POST` operation includes the message that was created in this collection of messages and a link to operate on that message.

Navigation Paths to messages

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums
      topics
         messages

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            topics
               messages
```

Supported Methods for messages

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters:** `startIndex`, `itemsPerPage` (pagination)
  - **response - body:** messages
- `POST`
  - **request - body:** message
  - **response - body:** message

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to From messages

Table 34–11 lists the resource types that the client can link to from this resource.

Table 34–11 Related Resource Types for messages

rel	resourceType
self	urn:oracle:webcenter:discussions:forum:topic:messages
	urn:oracle:webcenter:discussions:forum:topic:message

34.3.8.4.6 urn:oracle:webcenter:discussions:forum:topic:message Use this resource type to identify the URI to use to read (GET), update (PUT), and delete (DELETE) a specific discussion topic message. The response from a GET operation includes the specific message identified by the URI. The response from a PUT operation includes the modified version of the message identified by the URI. The response from a DELETE operation is a 204.

Navigation Paths to message

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   forums
      topics
         messages
            message

resourceindex
   spaces
      spaces:resourceindex
         spaces:forums
            topics
               messages
                  message
```

Supported Methods for message

The following methods are supported by this resource:

- GET
  - **request - body:** none
  - **response - body:** message
- PUT
  - **request - body:** message
  - **response - body:** message
- DELETE
  - **request - body:** none
  - **response - body:** none

Writable Elements for message

Table 34–12 lists the writable elements for this resource.

Table 34–12 Writable Elements for message

Element	Type	Required	Constraints	Description
`subject`	String	Yes	1 or more characters	Subject of this message
`body`	String	No	0 or more characters	Content of this message

Read-only Elements for message

Table 34–13 lists the read-only elements for this resource.

Table 34–13 Read-only Elements for message

Element	Type	Description
`id`	Integer	ID of the message
`parentId`	Integer	ID of the parent message
`forumId`	Integer	ID of the forum under which the message is posted
`topicId`	Integer	ID of the topic under which the message is posted
`createdBy`	String	ID of user that created the message
`author`	personReference	User information about the user that created the message, including GUID, ID, display name and a link to the profile icon (same user as `createdBy`)
`createdOn`	Date	Date on which the message was created
`updatedBy`	String	User that performed the last modification
`updatedOn`	Date	Date on which the message was last modified
`webUrl`	String	URL for direct access to the discussions server
`depth`	Integer	Depth in the hierarchy of messages
`messageCount`	Integer	Number of messages
`numberOfReplies`	Integer	Number of replies to this message
`hidden`	Boolean	True if this message is hidden
`hasAttachment`	Boolean	True if this message has attachments

Resource Types Linked to From message

Table 34–14 lists the resource types that the client can link to from this resource.

Table 34–14 Related Resource Types for message

rel	resourceType
self	`urn:oracle:webcenter:discussions:forum:topic:message`
	`urn:oracle:webcenter:discussions:forum:topic:messages`

34.3.9 Troubleshooting the Discussions Service

This section describes common problems and solutions for the Discussions service.

Problem

It can take several minutes to fetch data from recursive forums. For example, if you drop the Discussions task flow on the page with the `categoryId` and `forumId` parameters left blank and you set the `showrecursive` parameter to `${'true'}`, then it could take several minutes to scroll through all the forums.

Solution

The option to show recursive forums is not recommended for performance reasons.

Problem

Users can log on to WebCenter Portal but cannot log on to Oracle WebCenter Portal's Discussion Server.

Solution

Make sure that Oracle WebCenter Portal's Discussion Server is configured to use the same LDAP provider as the Framework application. Contact your administrator for details. For information about configuring LDAP, see the "Configuring the Identity Store" section in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Problem

When you access a discussions forum from your Framework application, the following error message is displayed:

```
No default or active connection available for: Discussion Forum
```

Solution

Ensure the following:

- Ensure that you have created a Discussion Forum connection.
- Ensure the required Discussion Forum connection is marked as the default connection.

Problem

While performing certain operations on Discussions Forums task flows, you encounter error messages like "Forum not found" or "Topic not found."

Solution

Check the following:

- Ensure that the discussions server is up and running.
- Ensure the forum or topic exist on the discussions server.
- Ensure that you have the required privileges for the forum or topic that you are editing.

If all of these settings are fine and the problem persists, contact your administrator for details.

Problem

If you change the connection to use a different discussions server, and if you change the application root category ID from administrator-services-discussions, then you could see exceptions like, "Category Not Found."

Solution

Restart the managed server on which the Framework application is deployed.

35

Integrating the Instant Messaging and Presence Service

This chapter explains how to integrate the Instant Messaging and Presence (IMP) service in a WebCenter Portal: Framework application at design time.

For more information about managing and including instant messaging and presence, see:

- "Managing the Instant Messaging and Presence Service" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Instant Messaging and Presence Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 35.1, "Introduction to the IMP Service"
- Section 35.2, "Basic Configuration for the IMP Service"
- Section 35.3, "Advanced Information for the IMP Service"

35.1 Introduction to the IMP Service

The IMP service enables you to observe the presence status (online, offline, busy, or away) of other authenticated application users. It provides instant access to interaction options, such as instant messages and mails. Additionally, if your enterprise presence is unavailable (for example, when you are traveling), you can connect to a 3rd-party network presence service, such as Yahoo! Messenger.

Services in WebCenter Portal that have user names with the same identity can integrate with the IMP service; for example, Discussions, Documents, or Mail.

This section provides an overview of IMP features and requirements. It includes the following subsections:

- Section 35.1.1, "Understanding the IMP Service"
- Section 35.1.2, "Requirements for IMP"

35.1.1 Understanding the IMP Service

Figure 35–1 shows the Presence icon indicating a user who is online.

Figure 35–1 Presence Icon (Online)



Wherever a user is indicated, for example as the author of a document in the document library, you can click the icon to invoke a context menu (Figure 35–2).

Figure 35–2 Presence Icon Context Menu



The context menu can include the following actions:

- View Profile (This opens the selected user's profile page, with information such as mail ID and contact numbers.)
- Send Mail (This opens a compose window for the mail client set, either WebCenter Portal's Mail service or a local mail client.)
- Change Credentials (This works as an alternative to using an external application, visible only to the current user.)
- Send Instant Message (This opens the instant message client running on the computer, Microsoft Communicator.)

Next to a contact name is an icon that indicates the presence state of each contact.

For detailed information about the IMP service at runtime, including screen shots and descriptions of the presence status options, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

35.1.2 Requirements for IMP

The IMP service requires a back-end presence server. WebCenter Portal is certified with Microsoft Office Live Communications Server (LCS) 2005, Microsoft Office Communications Server (OCS) 2007, and Microsoft Lync 2010.

Note: Oracle Beehive Server connections are not supported in this release.

35.2 Basic Configuration for the IMP Service

This section describes the steps required for adding the IMP service to your application. It includes the following subsections:

- Section 35.2.1, "Setting up Connections for the IMP Service"
- Section 35.2.2, "Adding the IMP Service at Design Time"
- Section 35.2.3, "Setting Security for the IMP Service"

35.2.1 Setting up Connections for the IMP Service

After the presence server is properly installed and running, you must add a connection to it. This section describes how. It includes the following subsections:

- Section 35.2.1.1, "IMP Service Connections"
- Section 35.2.1.2, "How to Set Up Microsoft LCS Connections for the IMP Service"
- Section 35.2.1.3, "How to Set Up Microsoft OCS and Microsoft Lync Connections for the IMP Service"

35.2.1.1 IMP Service Connections

The IMP service requires an **Instant Messaging and Presence** connection to the presence server (Microsoft LCS, Microsoft OCS, or Microsoft Lync).

When a service in WebCenter Portal interacts with an application that handles its own authentication, you can associate that application with an external application definition to allow for credential provisioning. An external application is mandatory.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

35.2.1.1.1 Mapping User Names to IM Addresses The `im.address.resolver.class` handles the resolver implementation used to map user names to IM addresses and IM addresses to user names. This implementation looks for IM addresses in the following places and order:

1. User Preferences - If the user has entered his or her IM address in the Presence service Preferences page, then the user's IM address is found in the user's preferences.
2. User Credentials - If an external application is configured, then an account field provides the user's IM address.
3. User Profiles - If the user has not supplied preferences and WebCenter Portal cannot fetch the IM address from the external application, then WebCenter Portal reads the IM address from LDAP (that is, the user's profile). The default LDAP property read is the `BUSINESS_EMAIL` attribute. Users can change this default with `im.address.profile.attribute`.

To use your own resolver implementation, extend from `IMPAddressResolver` class and implement two methods: `resolveAddress` and `resolveUsername`.

- The `resolveAddress` method takes in user name and returns the corresponding IM address.
- The `resolveUsername` method takes in the IM address and returns the corresponding user name.

To plug in the new resolver class, change the service property `im.address.resolver.class` from the default `oracle.webcenter.collab.rtc.IMPAddressResolverImpl` to your resolver implementation. Example 35–1 shows a sample `IMPAddressResolver` implementation, where the resolver appends the domain string `@example.com` to the

user name to construct the address and removes the same domain string from the address to construct the user name.

Example 35–1 Resolver Implementation

```
public class SampleAddressResolver extends IMPAddressResolver
{
  private String DOMAIN = "@example.com";

  public SampleAddressResolver()
  {
    super();
  }

  //Append DOMAIN to the username to construct the IM address
  public String resolveAddress(String username)
  {
    String imAddress = username;
    if(!imAddress.endsWith(DOMAIN))
    {
      imAddress = imAddress + DOMAIN;
    }

    return imAddress;
  }

  //Remove DOMAIN from the IM address to construct the username
  public String resolveUsername(String imAddress)
  {
    String username = imAddress;
    if(username.endsWith(DOMAIN))
    {
      int index = username.indexOf(DOMAIN);
      username = username.substring(0, index);
    }

    return username;
  }
}
```

See Also: Section, "setIMPServiceProperty" in *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*

35.2.1.2 How to Set Up Microsoft LCS Connections for the IMP Service

To set up the connection to the LCS presence server:

1. In Oracle JDeveloper, open the application in which to consume the Instant Messaging and Presence service.
2. In the Application Navigator, under Application Resources, right-click **Connections**, then select **Instant Messaging and Presence** from the list.
3. In the Create Instant Messaging and Presence connection dialog box, select to create the connection in **Application Resources**.

   A connection in Application Resources is available only for that application, while a connection in IDE connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE connections to avoid having to re-create it.

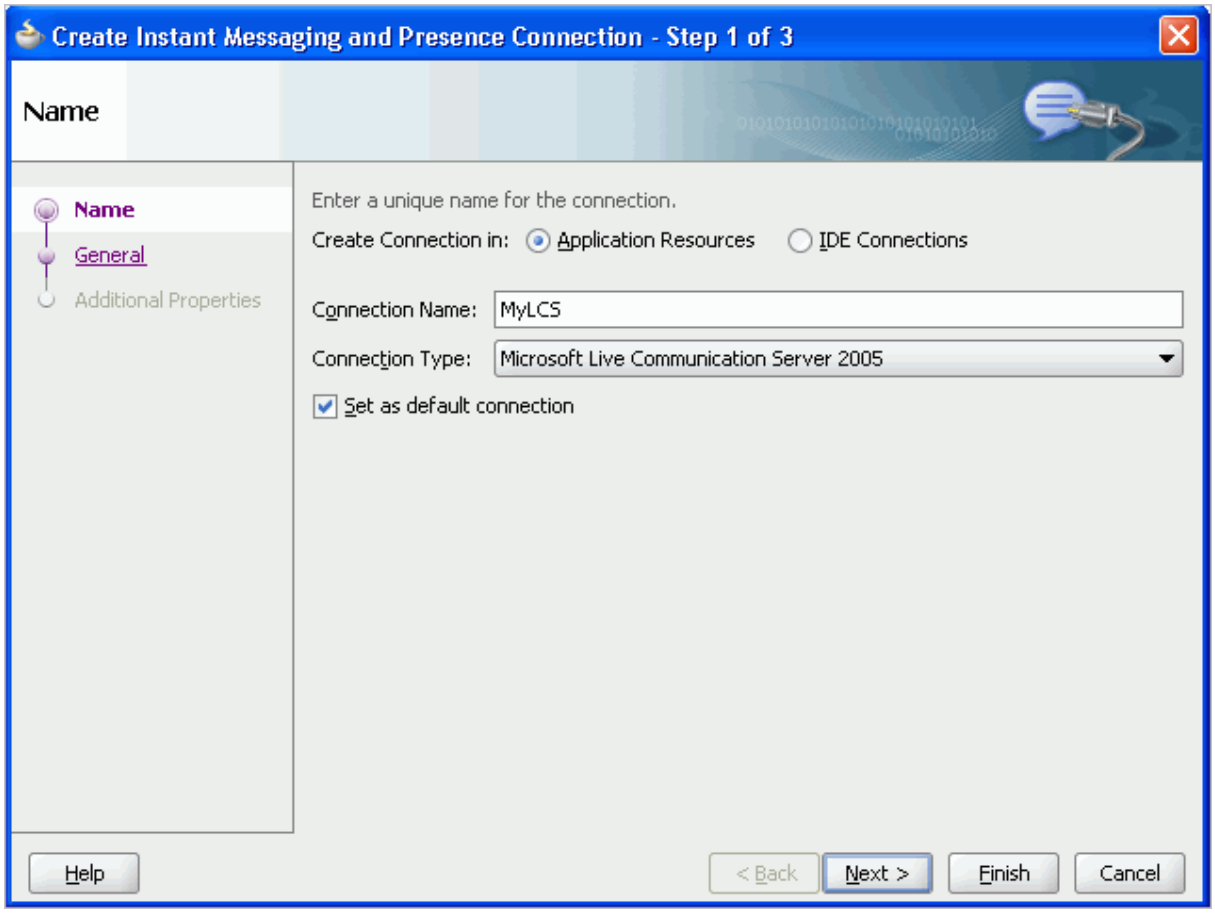
4. On the **Name** page, for **Connection Name**, enter a unique name for your connection.

   No other connection should have the same name.

5. From the **Connection Type** list, select **Microsoft Live Communication Server 2005**.

6. Select the **Set as default connection** checkbox to use this as the default connection, as shown in Figure 35–3.

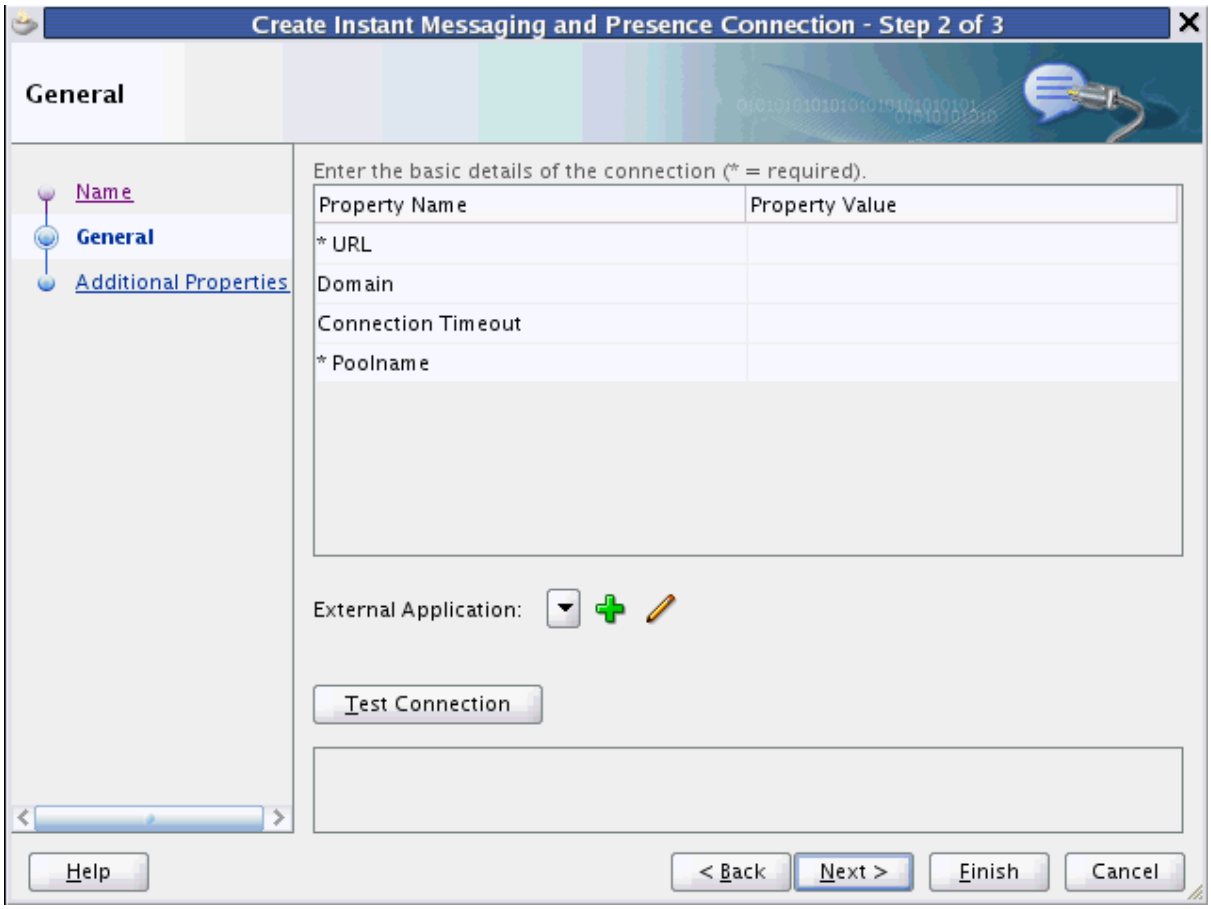
Figure 35–3 Create LCS Instant Messaging and Presence Connection, Step 1



7. Click **Next**

8. On the **General** page (Figure 35–4), enter the information for your Microsoft LSC instance.

Figure 35–4 Create LCS Instant Messaging and Presence Connection, Step 2



For example:

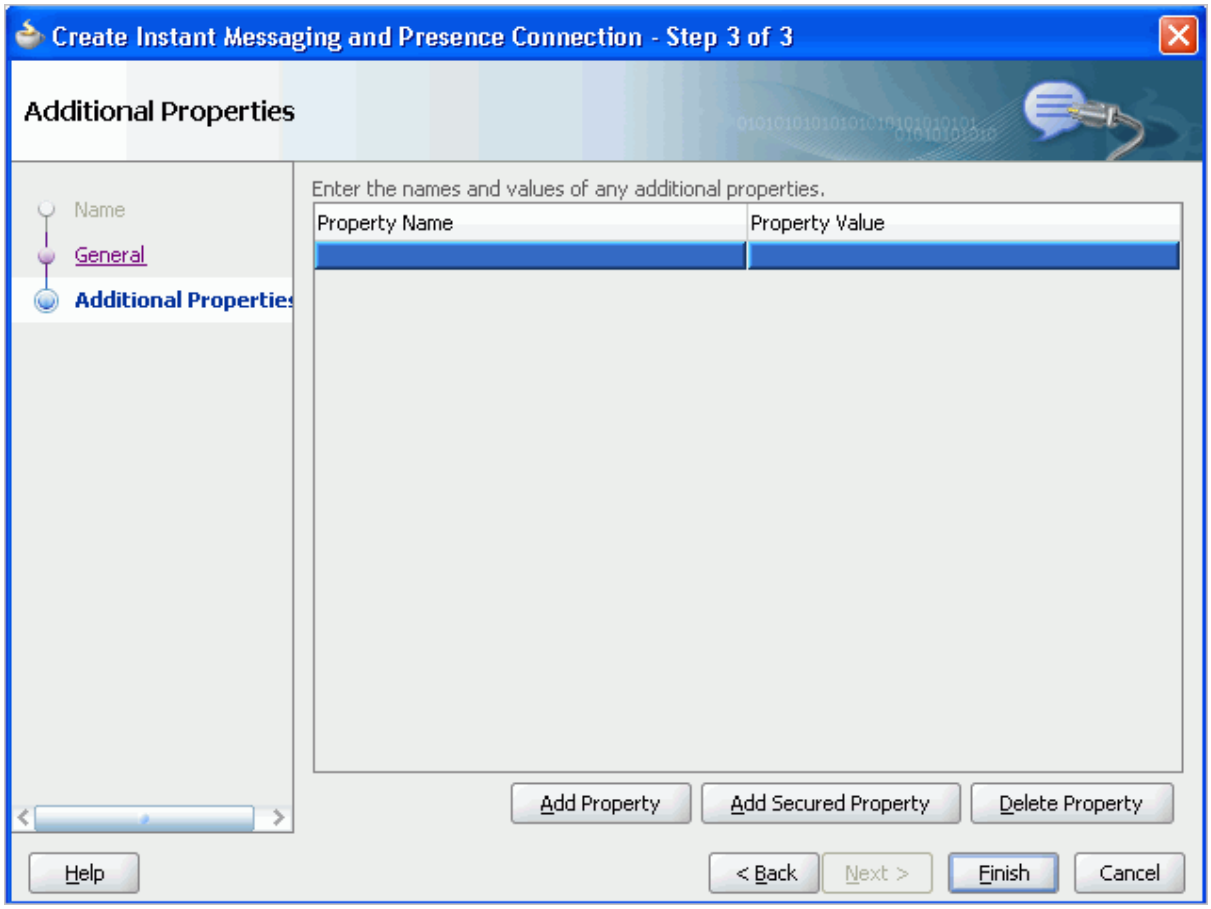
- The `Url` could be `http://host:port/RTC` where RTC is the virtual directory name under which the server side module is deployed. (See the Microsoft Live Communications Server 2005 documentation for more information.)
- The `Domain` property is maintained for backward compatibility. It should be left blank.
- The `Connection Timeout` property is optional. It represents the time (in seconds) the service should wait for the server to respond while making the connection. If the presence server does not respond in the given time, then it aborts the connection and reports an error.
- For `PoolName`, enter the name of the pool under which Microsoft Communications Server components are deployed. (See the Microsoft Live Communications Server documentation for more information.)

9. On the same page, select an **External Application** to leverage the authentication mechanism (user names and passwords) on the presence server.

   For LCS connections, an external application is mandatory. The application maps the presence server user to the application user such that end users do not have to enter their user names and passwords each time they need information. For detailed information about configuring an external application for the IMP service, see Section 35.2.3, "Setting Security for the IMP Service."

10. Click **Test Connection** to confirm that the connection is good.
11. Click **Next** to create the connection.
12. On the **Additional Properties** page (Figure 35–5), you can optionally add parameter.s

Figure 35–5 Create LCS Instant Messaging and Presence Connection, Step 3



13. Click **Finish**.

You can see the new IM and presence connection under **Application Resources - Connections**.

35.2.1.3 How to Set Up Microsoft OCS and Microsoft Lync Connections for the IMP Service

To set up the connection to the Microsoft OCS or Microsoft Lync presence server:

1. In Oracle JDeveloper, open the Framework application in which you plan to consume the Instant Messaging and Presence service.
2. In the Application Navigator, under Application Resources, right-click **Connections**, then select **Instant Messaging and Presence** from the list.
3. In the Create Instant Messaging and Presence connection dialog box, select to create the connection in **Application Resources**.

   A connection in Application Resources is available only for that application, while a connection in IDE connections is available for all applications you create. If you

plan to use the connection in other applications, then select IDE connections to avoid having to re-create it.

4. On the **Name** page, for **Connection Name**, enter a unique name for your connection.

   No other connection should have the same name.

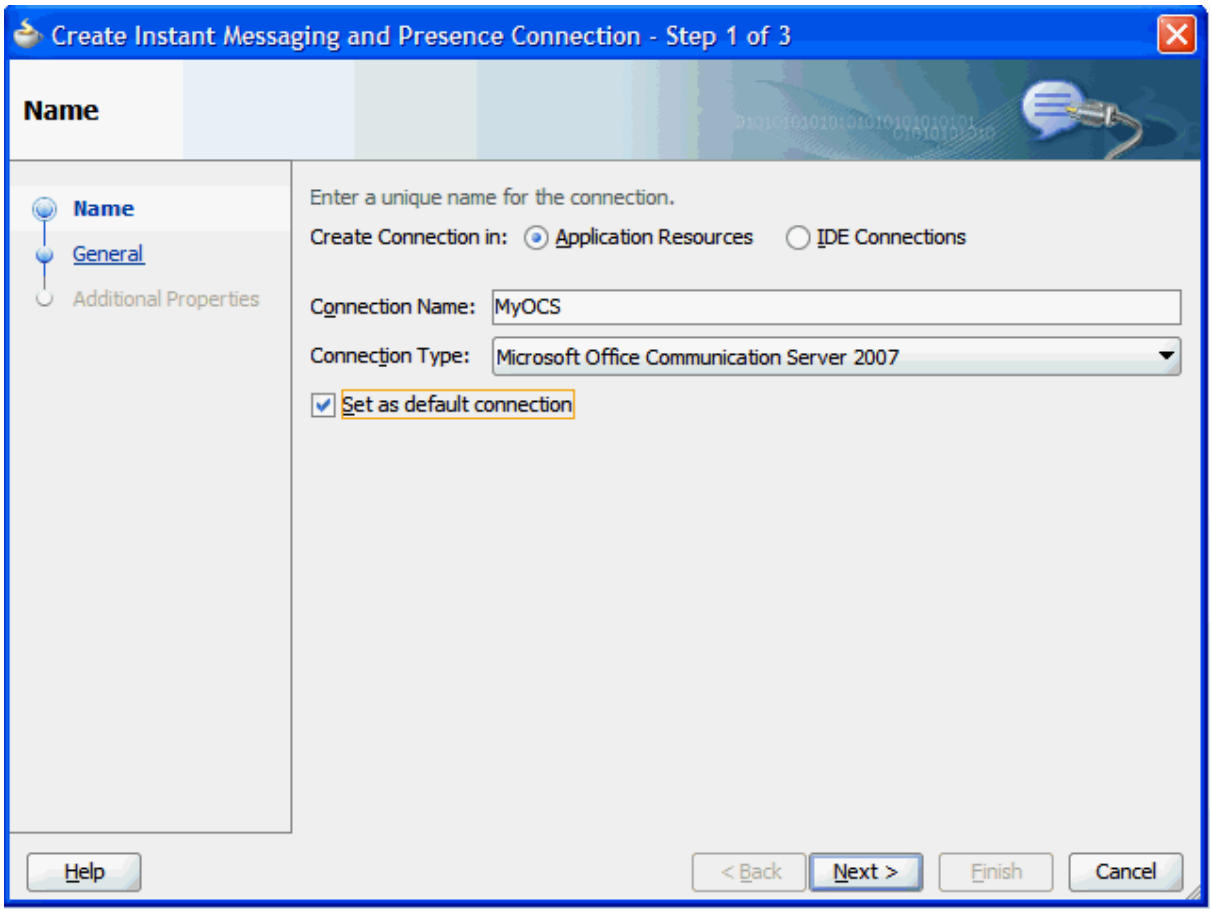
5. From the **Connection Type** list, select **Microsoft Office Communications Server 2007**.

   Note: Microsoft Lync connections use the Microsoft Office Communications Server 2010 connection type.

6. Select the **Set as default connection** checkbox.

   The service requires that one connection be marked as the default connection, as shown in Figure 35–6.

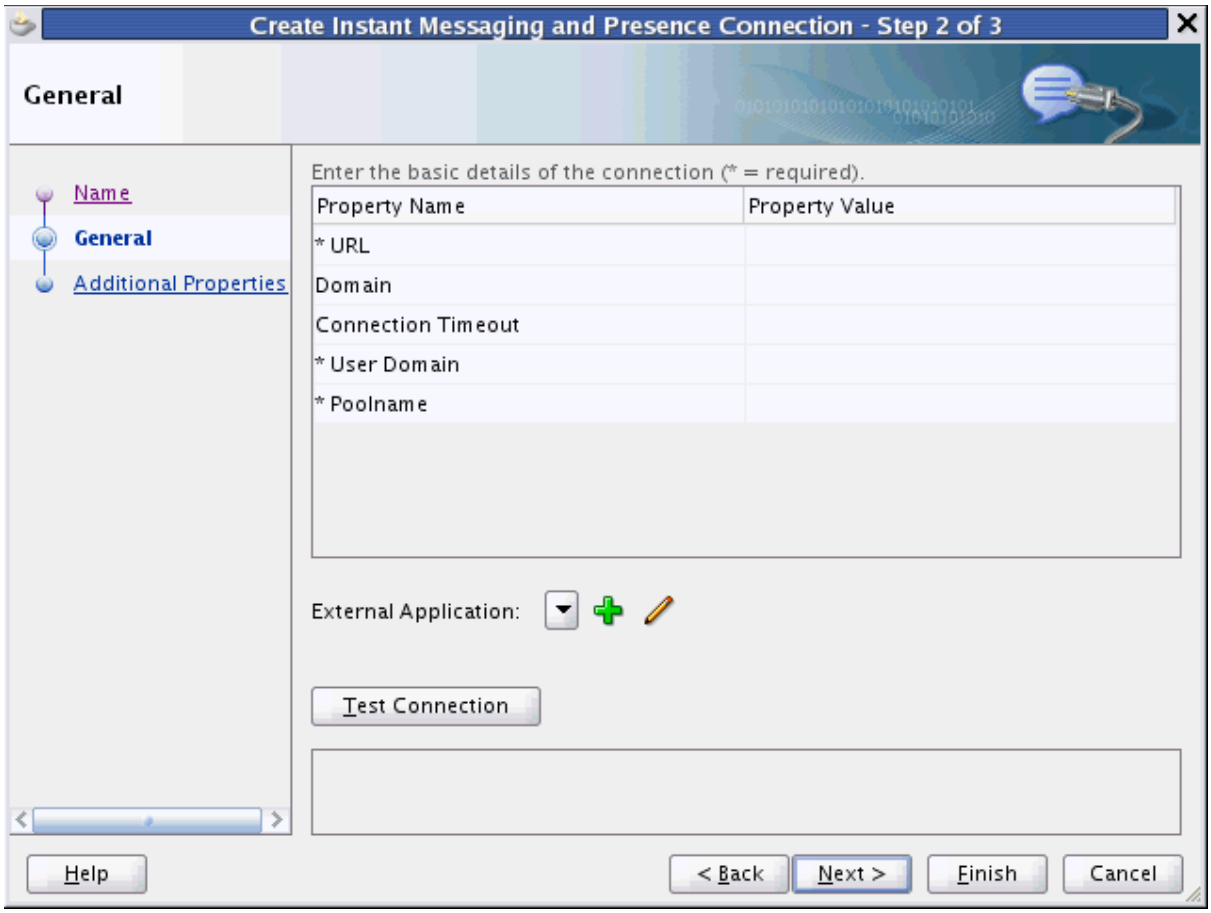
Figure 35–6 Create Microsoft OCS or Microsoft Lync IMP Connection, Step 1



7. Click **Next**.

8. On the **General** page (Figure 35–7), enter the parameters for your Microsoft OCS or Microsoft Lync instance.

Figure 35–7 Create Microsoft OCS or Microsoft Lync IMP Connection, Step 2



For example:

- The `URL` is the location of your Microsoft OCS or Lync instance. This could be `http://host:port/RTC` where RTC is the virtual directory name under which the server side module is deployed. (See the Microsoft documentation for more information.)
- The `Domain` property is maintained for backward compatibility. It should be left blank.
- The `Connection Timeout` property is optional. It represents the time (in seconds) the service should wait for the server to respond while making the connection. If the presence server does not respond in the given time, then it aborts the connection and reports an error.
- The `User Domain` is the Active Directory domain on Microsoft OCS or Lync. This parameter is mandatory.
- For `Poolname`, enter the name of the pool under which Microsoft Communications Server components are deployed. This parameter is mandatory. (See the Microsoft documentation for more information.)

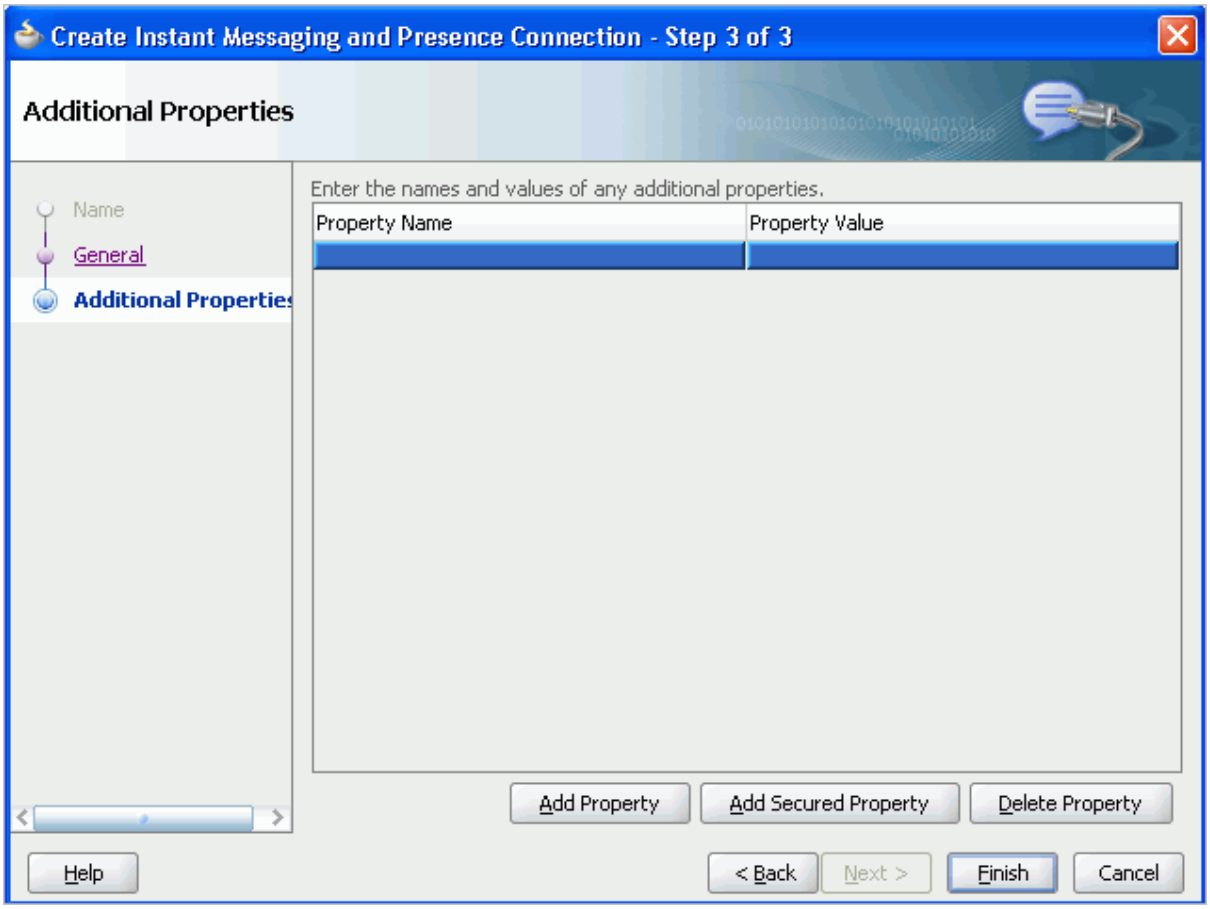
9. On the same page, select an **External Application** to leverage the authentication mechanism (user names and passwords) on the presence server.

   For Microsoft OCS and Microsoft Lync connections, an external application is mandatory. The application maps the presence server user to the application user such that end users do not have to enter their user names and passwords each time they

need information. For detailed information about configuring an external application for the IMP service, see Section 35.2.3, "Setting Security for the IMP Service."

10. Click **Test Connection** to confirm that the connection is good.
11. Click **Next**.
12. On the **Additional Properties** page (Figure 35–5), you can optionally add parameters.

Figure 35–8 Create Microsoft OCS or Microsoft Lync IMP Connection, Step 3



13. Click **Finish**.

You can see the new IM and presence connection under **Application Resources - Connections**.

35.2.2 Adding the IMP Service at Design Time

This section explains a basic incorporation of the IMP service into your application. It includes the following subsections:

- Section 35.2.2.1, "IMP Service Task Flows"
- Section 35.2.2.2, "How to Add the IMP Service to your Application"

35.2.2.1 IMP Service Task Flows

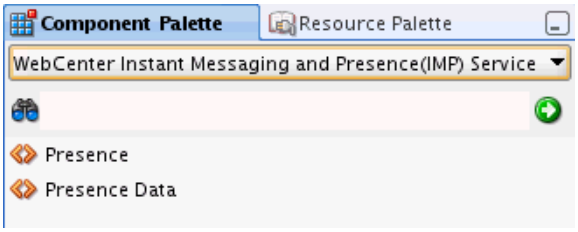
The IMP service does not include any task flows.

35.2.2.2 How to Add the IMP Service to your Application

To add the IMP service to your Framework application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application.
2. Open the page on which you want to add the IMP service.
3. Ensure that you have configured your application to connect to the presence server. See Section 35.2.1, "Setting up Connections for the IMP Service."
4. In the Component Palette, drag and drop the **Presence** component to the page to add the Presence icon (Figure 35–9).

Figure 35–9 Component Palette - IMP Service Components



Use the **Presence** icon anywhere you want to display a user, for example, as the author of a discussion topic, the sender/recipient of a mail, or the owner of a document.

5. In the resulting dialog box, enter the user name of a user that exists in the back-end server that is linked to in the IM and Presence connection.

   You can add numerous presence components to the application page.

   For information about optional **Presence** component parameters, see Section 35.3.2, "Customizing IMP Views."

6. Click **Finish**.
7. From the Component Palette, drag and drop **Presence Data** to the end of the page (that is, before the `<af:document>` tag).

   This component does not have any attributes.

   The **Presence Data** component provides the status information for all **Presence** components on the page, such as online, offline, or busy. It verifies that all presence information corresponding to the user on the whole page shows consistent status information. Without this component, all users appear offline.

   Because the **Presence Data** component makes a call to the back-end server, for best performance, ensure that this is the last tag on the page. To avoid adding this tag to every page in your application, consider using a page template with **Presence Data** as the last component; that is, before the end of the `</af:form>` tag.

> **Note:** You can create new pages at runtime on which you can add components, such as forums, mail, or documents. Many components have **Presence** tags, but users do not have a handle to add the **Presence Data** tag to the page. To see presence on custom pages, you must manually add the **Presence Data** tag to the underlying template.

8. If **External Application** in IDE connections was selected when you created the connection, then drag and drop the **External Application - Change Password** task flow into your application from the Resource Palette or Component Palette.

   This task flow enables the end user to set the appropriate user name and password for the external application.

9. Save your project, and then run your page to a browser to see the Presence component.

   Figure 35–10 shows the runtime Presence component with the display name of user Monty Montasaurus111.

Figure 35–10 WebCenter Portal Presence Icon - Online



35.2.3 Setting Security for the IMP Service

The IMP service requires user identity. ADF security is configured by default if you created your application using the WebCenter Portal - Framework Application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

Credentials are read from the external application (public credentials) and used to log on to the presence server. If you do not apply ADF security or if you do not have an external application configured, then users cannot authenticate and do not see any content at runtime.

> **Note:** The presence server and the Framework application should point to the same identity store. The identity store must be LDAP-based; that is, not file-based with `jazn-data.xml`.

To access the presence server, the IMP service can use an external application with dedicated user accounts.

Microsoft LCS, OCS, and Lync support external application connections. With a secured application, users get presence status. The Change Credentials option works as an alternative to using an external application. Logged-in users can click their own Presence context menu and select **Change Credentials**. Security should be on a private trusted network.

To use an external application for authentication:

1. On the General page of the Create Instant Messaging and Presence connection wizard, click the **+** icon next to External Application.

   This brings up the Register External Application wizard. The application maps the presence server user to the application user such that end users do not have to enter their user names and passwords each time.

   > **Note:** External application credential provisioning is built into the IMP connection. You do not need to drop **External Application - Change Password** task flow on a page.

2. On the Name page:

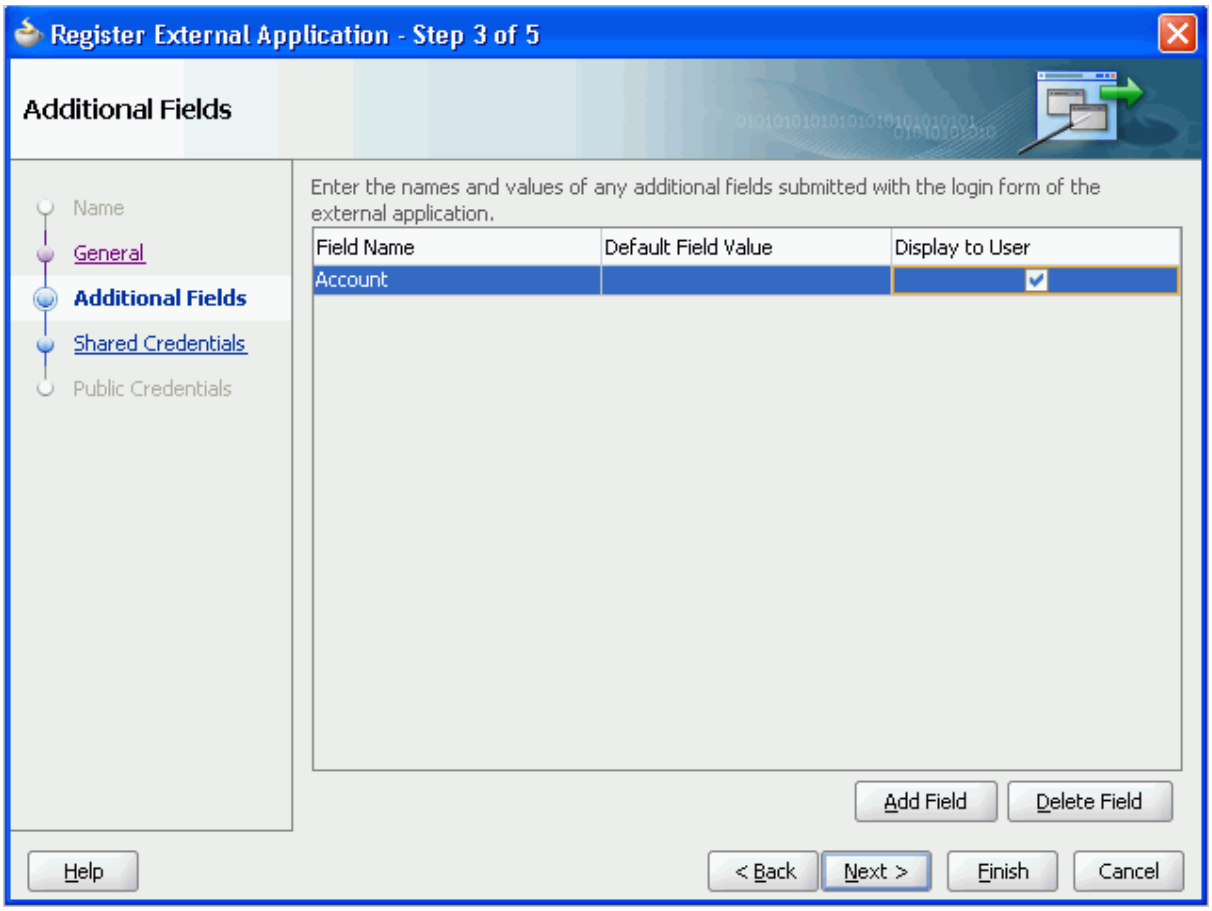
- For **Application Name**, enter a unique name to identify the application. This name must be unique within the Framework application, and among other connections as well. Note that you cannot edit this field afterward.
- For **Display Name**, enter a name for the application that end users see in the credential provisioning screens.

3. Click **Next**.
4. On the General page, leave the following properties with the default values.
   - **Login URL**
   - **User Name/ID Field Name**
   - **Password Field Name**
5. From the **Authentication Method** list, select **POST**. This submits login credentials within the body of a form. The external application for the IMP service requires this authentication method.
6. Click **Next**.
7. On the Additional Fields page:

   Click **Add Field**, and add an extra field with the name "Account." Make sure to select the **Display to User** checkbox, as shown in Figure 35–11.

   Note: The external application for the IMP service requires this additional field. It must be displayed to users.

Figure 35–11 Account Additional Field



8. The External Application service allows different types of credentials to be associated with a connection:
   - When shared credentials are specified, every *authenticated* user uses the same credentials to access the external application; that is, the user name and password you can define here. A single presence session is created for all logged-in users. This is not accessible to public users.
   - With public credentials, without authenticating your Framework application, you can view presence from a certain presence ID for all *unauthenticated* (public) users. Public credentials are used whenever an application is not secured or the user has not yet logged in. A single presence session is created for all public users.
   - With private credentials, each user must authenticate to an *individual* ID (that is, each application user must specify his own credentials). One presence session is created for each user.
9. Click **Finish** to have the external application use private credentials, or click **Next** to set up shared or public credentials.
10. *For Shared Credentials Only*: On the Shared Credentials page, ensure that **Specify Shared Credentials** is selected, then enter the shared user credentials and ID.
11. *For Public Credentials Only*: On the Public Credentials page, ensure that **Specify Public Credentials** is selected, then enter the user credentials and ID for public use.

12. Click **Finish** to register the external application.
13. In the IMP connection wizard, ensure that this newly-created external application connection for IMP is selected.

For information about using external applications, see Section 69.13, "Working with External Applications."

35.3 Advanced Information for the IMP Service

This section describes optional features available with the IMP service. It includes the following subsections:

- Section 35.3.1, "Enabling Network Presence"
- Section 35.3.2, "Customizing IMP Views"
- Section 35.3.3, "Troubleshooting the IMP Service"

35.3.1 Enabling Network Presence

When WebCenter Portal presence is not available (for example, if your enterprise uses a Jabber/XMPP presence server or has federated presence servers with users distributed across identity management systems), you can connect to a 3rd-party network presence service.

Out-of-the-box, WebCenter Portal supports Yahoo! Messenger on network presence. However, the network presence model can be extended to include other providers, such as ICQ.

This section includes the following subsections:

- Section 35.3.1.1, "Setting Up Yahoo! Messenger Presence"
- Section 35.3.1.2, "Setting Up Other Network Presence Providers"

35.3.1.1 Setting Up Yahoo! Messenger Presence

WebCenter Portal Framework allows end users to set their IM preferences to their Yahoo! Messenger presence. Portal developers enable this functionality by leveraging the `rtcPresenceHandler` bean.

Follow these steps to include Yahoo! Messenger presence in your Framework application.

1. On the Source tab for your JPSX page, add the user interface for users to enter their Yahoo! Messenger credentials: **Display Name** and **IM Address** input boxes and a **Save IM Preferences** button.

   Example 35–2 shows the two `af:inputText` components for users to enter Display Name and IM Address and an `af:commandButton` component for users to save the preferences.

   Setting the `inputText` value with this EL sets the `displayName` and `imAddress` in the preferences bean. To save the values, users click the button, which invokes a method from the bean to save the preferences.

Example 35–2 User Interface for Yahoo! Messenger Presence

```
      <af:inputText label="Display Name"
value="#{rtcPreferenceHandler.displayName}" id="it1"/>
      <af:inputText label="IM Address"
value="#{rtcPreferenceHandler.imAddress}" id="it2"/>
```

```
                <af:commandButton text="Save IM Preferences" id="cb1"
            actionListener="#{rtcPreferenceHandler.savePreferences}"/>
```

Example 35–3 shows the full source code for the page, with the user interface included.

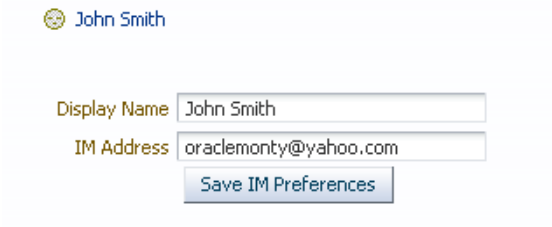
Example 35–3 Source Code for the Page with Yahoo Presence

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:rtc="http://xmlns.oracle.com/webcenter/collab/rtc">
  <jsp:directive.page contentType="text/html;charset=UTF-8"/>
  <f:view>
    <af:document id="d1">
      <af:form id="f1" usesUpload="true">
        <af:panelStretchLayout id="psl1">
          <f:facet name="bottom"/>
          <f:facet name="center">
            <af:panelGroupLayout layout="scroll"
                                 xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
                                 id="pgl1">
              <rtc:presence username="#{rtcPreferenceHandler.imAddress}"
displayName="#{rtcPreferenceHandler.displayName}" id="p1"/>
              <rtc:presenceData id="pd1"/>
              <af:inputText label="Display Name" value="#{rtcPreferenceHandler.displayName}"
id="it1"/>
              <af:inputText label="IM Address" value="#{rtcPreferenceHandler.imAddress}" id="it2"/>
              <af:commandButton text="Save IM Preferences" id="cb1"
                                actionListener="#{rtcPreferenceHandler.savePreferences}"/>

            </af:panelGroupLayout>
          </f:facet>
          <f:facet name="start">
            <af:spacer width="10" height="10" id="s1"/>
          </f:facet>
          <f:facet name="end">
            <af:spacer width="10" height="10" id="s2"/>
          </f:facet>
          <f:facet name="top"/>
        </af:panelStretchLayout>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

2. Run the JPSX page to see that now users can enter their Yahoo! Messenger credentials (Figure 35–12).

Figure 35–12 User Interface to Enter Yahoo! Messenger Credentials



Presence tags for that user shows their Yahoo presence, either online (Figure 35–13) or offline (Figure 35–14).

Figure 35–13 Yahoo Presence Icon - Online

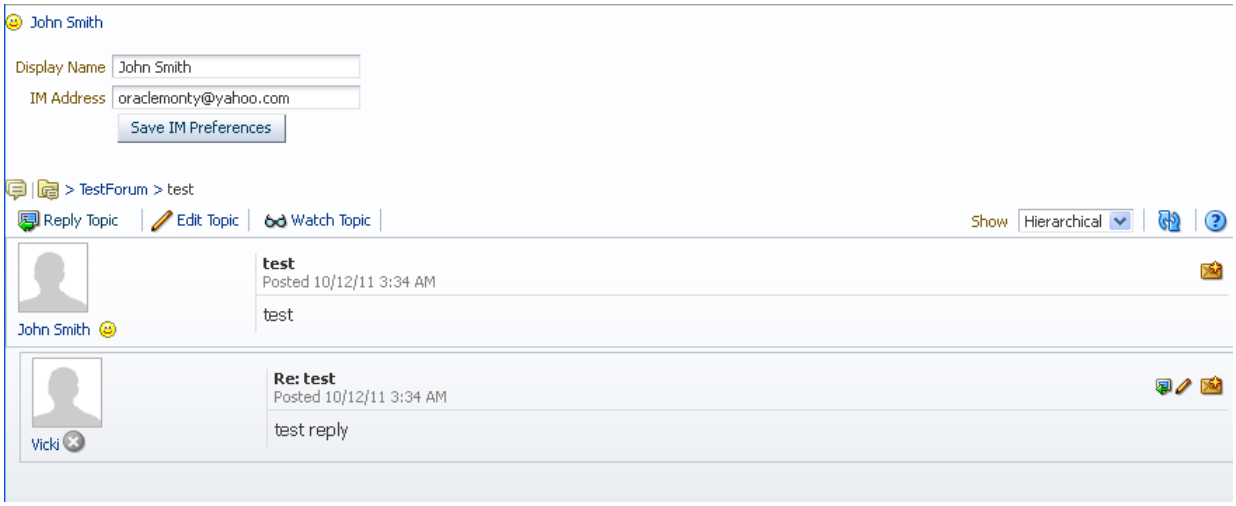


Figure 35–14 Yahoo Presence Icon - Offline



Figure 35–15 shows a sample page that includes the Discussion Forums task flow, where Yahoo! Messenger presence and WebCenter Portal presence are shown together.

Figure 35–15 Sample Page with WebCenter Portal Presence and Yahoo! Messenger Presence



35.3.1.2 Setting Up Other Network Presence Providers

To use a different network presence provider, deploy your application that contains the `PresenceNetworkAgent` implementation class for that service and an `imp-pna.config` file listing that implementation class into a jar file.

For example, suppose the new Presence Network Agent is called SamplePNA. You must create two files: the `Project1.SamplePNA` class file and the `imp-pna.config` file.

- `Project1.SamplePNA.java`: This class implements `oracle.webcenter.collab.rtc.PresenceNetworkAgent`. It must implement three methods:
  - `isSupported(PNAContext context)`: The method should return true if the PNA supports the user for which presence is requested. The `PNAContext` object supplied contains the user information. For example, if the PNA supports all IM addresses with the domain `example.com`, then it can get the imAddress from the `PNAContext` object and check for the domain `example.com`.
  - `getURL(PNAContext context)`: The method should return the fully qualified URL to reach to the user's Presence icon. Again, it gets the user information from the `PNAContext` object.
  - `getChatURI(PNAContext context)`: The method should return the browser compatible URI to invoke the chat client (for example, `sip:user@example.com`).

  Example 35–3 shows this SamplePNA class file.

Example 35–4 SamplePNA Class File

```
public class SamplePNA implements PresenceNetworkAgent
{
private String DOMAIN = "@example.com";

public SamplePNA()
{
super();
}

//Returns true if the imAddress ends with DOMAIN
public Boolean isSupported(PNAContext context)
{
String imAddress = context.getIMAddress();
if(imAddress.endsWith(DOMAIN))
{
return true;
}

return false;
}

//Returns the URL to the icon representing the current status of a user
public String getURL(PNAContext context)
{
String imAddress = context.getIMAddress();
Strung url = "http://www.example.com?address=" + imAddress;
return url;
}

//Returns the browser compatible chat uri to invoke the thick client
public String getChatURI(PNAContext context)
{
String imAddress = context.getIMAddress();
String chatURI = "sip:" + imAddress;
return chatURI;
}
}
```

- `META-INF/imp-pna.config`: This file lists the available `PNAAgent` classes. In this case, it contains only the following line:

  ```
  Project1.SamplePNA
  ```

See Chapter 56, "Extending the Spaces Application Using JDeveloper" for information on how to package your application and deploy it as a custom shared library (for example, in WebCenter Portal: Spaces).

Whenever a Presence tag is encountered with an IM address as *`user@example.com`*. The URL configured appears on the user interface as the Presence icon.

```
<rtc:presence username="user1@example.com" resolveAddress="false"/>
```

35.3.2 Customizing IMP Views

Table 35–1 lists the attributes supported by the **Presence** component. Only the `username` attribute is required; all other attributes are optional. You can update these attributes in the Property Inspector.

Table 35–1 Presence Component Description

Attribute	Description
`username`	The user whose presence information you want to add to the application page. This attribute is required.
`display`	How the component should display. Takes one of the following values: ■ `icon`: Display only the Presence icon; do not render the name. ■ `name`: Display the user name; do not display the Presence icon. ■ `both` (default): Display both the icon and the user name.
`displayName`	By default, the **Presence** component displays the user name. If the flag `get.display.name.from.user.profile` is set to `true` in `service-config.xml` *and* if this `displayName` attribute is not supplied, then the **Presence** component tries to look up a user's display name from the User Profile.
`iconPosition`	The position for the icon. Possible values are `begin` and `end`. The default value is `begin`.
`controlsEnabled`	A Boolean value that defines whether the component should provide rich interactions to the end user. If this attribute is set to `false`, then the Presence component does not do anything when clicked.
`id`	A unique identifier for the component on the page. The identifier must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Z, a-z) or an underscore ('_'). ■ Subsequent characters must be an ASCII letter or digit (A-Z, a-z, 0-9), an underscore ('_'), or a dash ('-').
`smallIcon`	A Boolean value that defines whether to use the small 12x12 icon set (`true`) or to use the default 16x16 icon set (`false`). The default value is `false`.
`rendered`	A Boolean value that defines whether to render this component. The default value is `true`.
`binding`	An EL reference to store the component instance on a bean. Use this to give programmatic access to a component from a backing bean or to move creation of the component to a backing bean.

Table 35–1 (Cont.) Presence Component Description

Attribute	Description
`inlineStyle`	The CSS styles to use for this component. Manually enter any style in compliance with CSS version 2.0 or later, or expand this node to specify style elements. This is intended for basic style changes.
`styleClass`	A CSS style class to use for this component.
`resolveToAddress`	A Boolean value that defines whether to resolve the supplied `username` to an IM address. The default value is `true`. However, if the supplied `username` is an IM address, then you can set this value to `false`, and the IMP service uses `username` as the IM address.

35.3.3 Troubleshooting the IMP Service

This section describes common problems and solutions for the IMP service.

Problem

The Presence icon is not visible in your Framework application.

Solution

Ensure that an IMP connection exists in your application and has been set as active.

Problem

Changes in the presence status of users are not visible in your Framework application.

Solution

For each logged-in user's session, the IMP service fetches the presence information from the presence server and stores it in the presence cache. For presence requests, the service returns the data from the cache until the cache expires. The default cache expiry period is 60 seconds.

To view the updated presence status, you can wait for the cache to expire and retrieve the latest presence status.

You can also change the cache expiry time by setting the `rtc.cache.time` service configuration property to the desired value (in seconds). Update `adf-config.xml` to include the highlighted entry, which shows the sample value as 30 seconds. Example 35–5 shows an example.

Example 35–5 Setting the rtc.cache.time Expiration Value in adf-config.xml

```
<adf-collaboration-config xmlns="http://xmlns.oracle.com/webcenter/collab/config">
<service-config serviceId="oracle.webcenter.collab.rtc">
<property name="rtc.cache.time" value="30"/></service-config>
</adf-collaboration-config>
```

Problem

A number of options, such as Send Instant Message, are not available in the context menu of the IMP service in your Framework application.

Solution

Ensure that IMP service is configured in your Framework application. Also, ensure the following settings for the various context menu options:

- **View Profile** option unavailable: Ensure that your application is secured.
- **Send Instant Message** option unavailable: Ensure that the IMP service is configured in the application.

Problem

You are unable to send a message from your Framework application. Clicking the **Send Instant Message** option returns an error.

Solution

Ensure that your SIP client is supported by the presence server and you have logged on as an authenticated user. The supported SIP client is Microsoft Communicator.

36

Integrating the Mail Service

This chapter explains how to integrate the Mail service in a WebCenter Portal: Framework application at design time.

For more information about managing and including mail, see:

- "Managing the Mail Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Mail Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 36.1, "Introduction to the Mail Service"
- Section 36.2, "Basic Configuration for the Mail Service"
- Section 36.3, "Advanced Information for the Mail Service"

> **Note:** The Mail service supports only the Inbox. No other folders (or moving of messages) are supported.

36.1 Introduction to the Mail Service

This section provides an overview of Mail service features and requirements. It includes the following subsections:

- Section 36.1.1, "Understanding the Mail Service"
- Section 36.1.2, "Requirements for the Mail Service"

36.1.1 Understanding the Mail Service

The Mail service enables users to access the inbox of a mail server that supports Internet Message Access Protocol4 (IMAP4) and Simple Mail Transfer Protocol (SMTP). Additionally, it enables users to compose a new mail message from within the application (with attachments) and delete, reply to, and forward messages.

The Mail service does not replace users' mail clients; it simply enables users to access and compose mail in a single, collaborative environment.

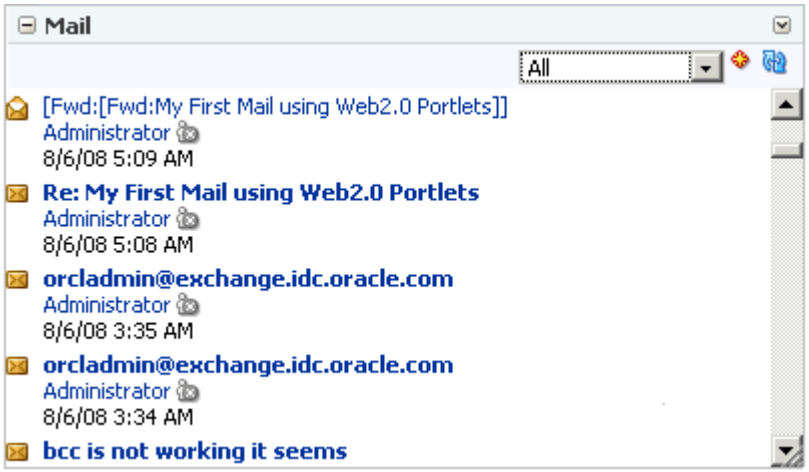
With the Mail service, you can do the following:

- Read a message by clicking the linked mail subject.
- View sender details (including date) by expanding the **From** field.

- Scroll down to see the other messages not in the view. You can navigate among the fetched messages as cached messages.
- Attach a file to a message by expanding the attachment section within the mail dialog and clicking **Attach**. Specify an attachment in the new dialog pop-up. Remove an attachment from the mail by clicking the **Remove Attachment** icon.
- Reply to a message by clicking the **Reply** or **Reply All** icon.
- Forward a message by clicking the **Forward** icon.
- Cancel an operation (for example, sending a mail) by clicking **Cancel**.

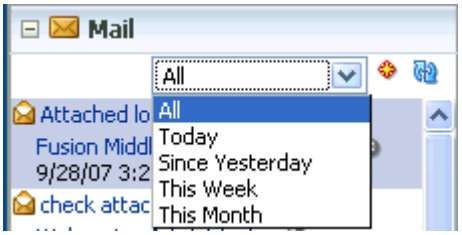
Figure 36–1 shows the Mail task flow at runtime. At the top of the view are three elements: a dropdown list that shows the number of mails to display (here, **All**), the **Compose** icon, and the **Refresh** icon. The **Refresh** icon provides a means of manually checking for new messages in the inbox.

Figure 36–1 Mail Service at Runtime



The dropdown list provides filters to focus the view on messages received today, messages received since yesterday, messages received this week, and messages received this month (Figure 36–2).

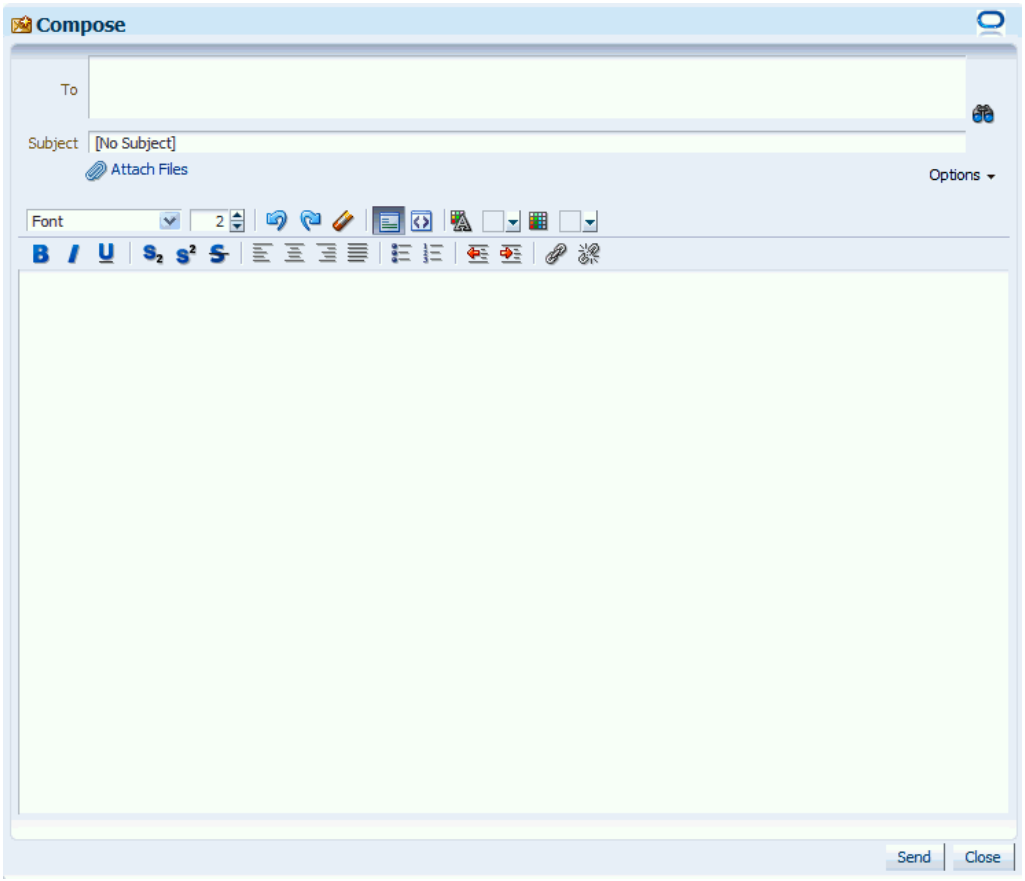
Figure 36–2 Message Filter



Note: By default, `All` displays the 50 most recent messages. For information on how to increase this amount, see Section 36.3.2, "Configuring the Number of Mails Displayed."

Click the **Compose** icon next to the dropdown list to compose a new message right from your application. Clicking this icon displays the Compose page, as shown in Figure 36–3.

Figure 36–3 Compose Mail



Use the **Search** icons to find mail addresses and contacts of users in the LDAP store that the Framework application uses. For any user not in the LDAP store, you must enter an explicit mail address.

For more information about the Mail service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

36.1.2 Requirements for the Mail Service

The Mail service requires a mail server that supports IMAP4 and SMTP protocols.

In WebCenter Portal: Spaces, the Microsoft Exchange mail server is required for automatic creation of distribution lists when spaces are created. In a Framework application, this feature may not be desirable. To disable it, do not provide the LDAP (Active Directory) server details in the mail connection.

36.2 Basic Configuration for the Mail Service

This section describes the steps required for adding the Mail service to your application. It includes the following subsections:

- Section 36.2.1, "Configuration Roadmap - Mail Service"

- Section 36.2.2, "Setting up Connections for the Mail Service."
- Section 36.2.3, "Adding the Mail Service at Design Time"
- Section 36.2.4, "Setting Security for the Mail Service"

36.2.1 Configuration Roadmap - Mail Service

Use the roadmap in this section as a guide through the configuration process.

Figure 36–4 and Table 36–1 provide an overview of the prerequisites and tasks required to get the Mail service working in Framework applications.

Figure 36–4 Configuring the Mail Service for Framework Applications

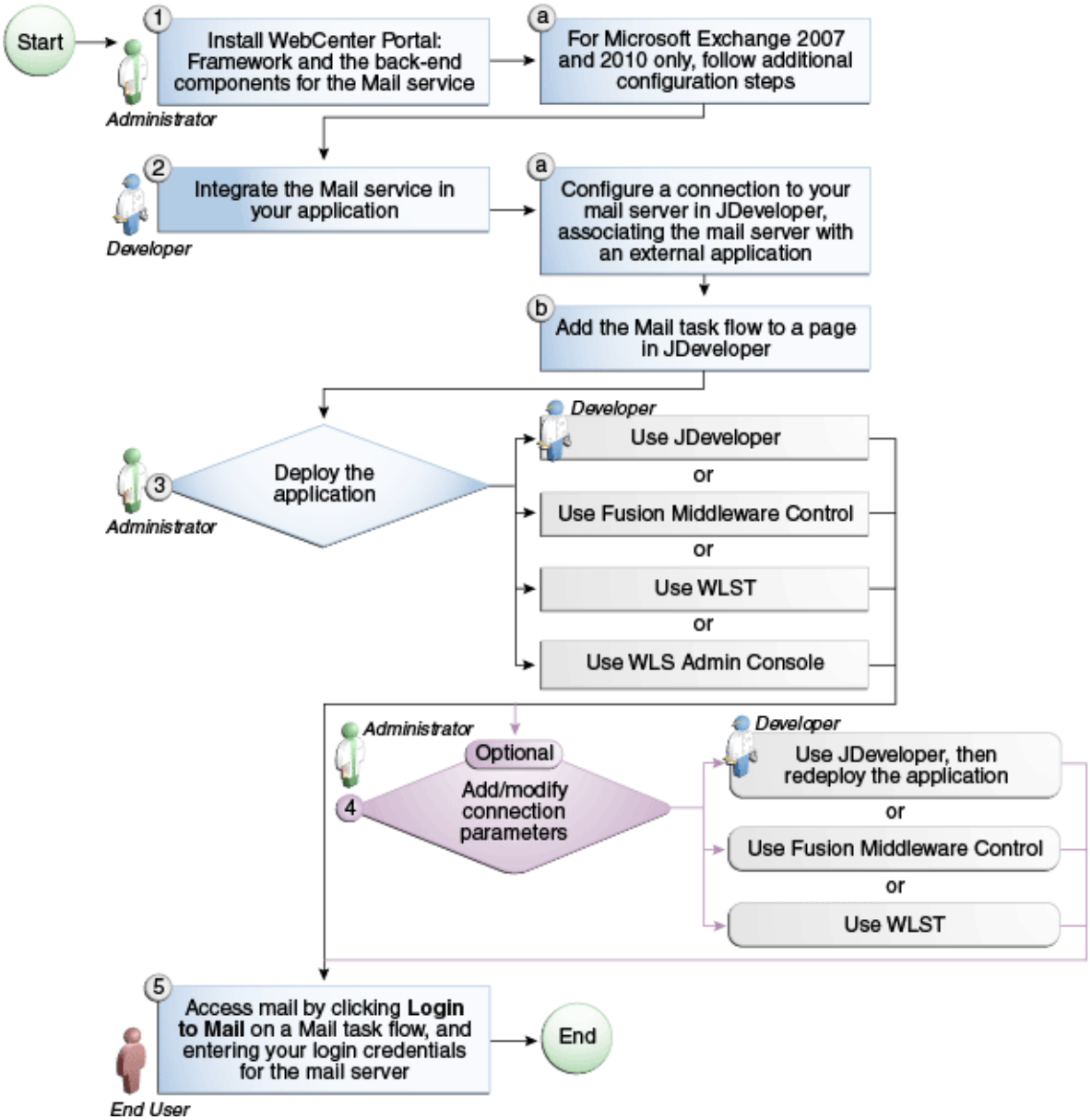


Table 36–1 Configuring the Mail Service for Framework Applications

Actor	Task	Sub-task
Administrator	1. Install WebCenter Portal and the back-end components for the Mail service	1.a For Microsoft Exchange 2007 and 2010 only, follow additional configuration steps
Developer	2. Integrate the Mail service in your Framework application	2.a Configure a connection to your mail server in JDeveloper, associating the mail server with an external application
		2.b Add the Mail task flow to a page in JDeveloper
Developer or Administrator	3. Deploy the Framework application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Control (Administrator)	
Developer or Administrator	4. (Optional) Add/modify connection parameters using one of the following tools: ■ JDeveloper, then redeploy the application (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator)	
End User	5. Access mail by clicking **Login to Mail** on a Mail task flow, and entering your login credentials for the mail server	

36.2.2 Setting up Connections for the Mail Service

Before you can use the Mail service, you must first set up the connection to your mail server. The Mail service supports any mail server based on IMAP4 and SMTP protocol.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To create a connection to your mail server from the application:

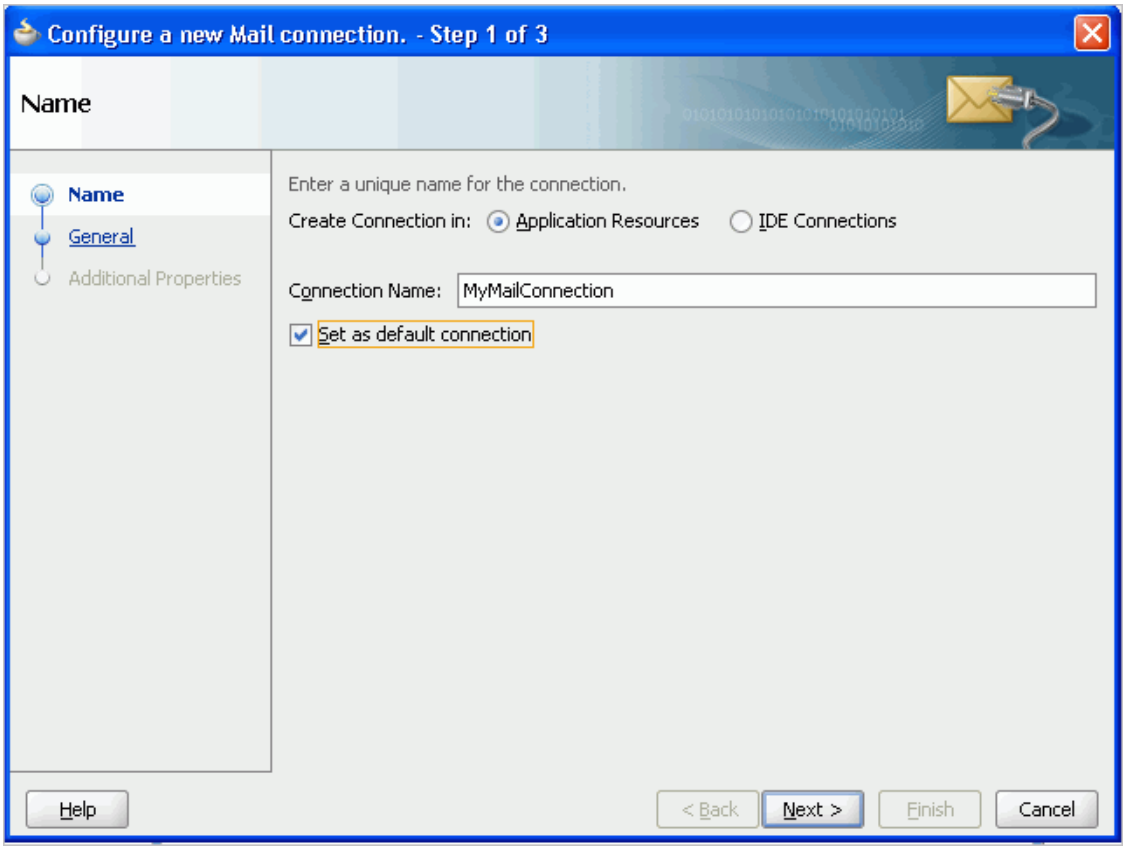
1. In Oracle JDeveloper, open the application in which to consume the Mail service.
2. In your application, under Application Resources, right-click **Connections**, then select **Mail** from the list.
3. Select to create the Mail service connection in **Application Resources**.

   A connection in Application Resources is available only for that application; while a connection in IDE Connections is available for all applications you create. If you

plan to use the connection in other applications, then select IDE Connections to avoid having to re-create it.

4. In the **Connection Name** field, enter a unique name for the connection.
5. When configuring a single mail account, select the **Set as default connection** checkbox to use this as the active connection (Figure 36–5).

Figure 36–5 Configure a New Mail Connection, Step 1



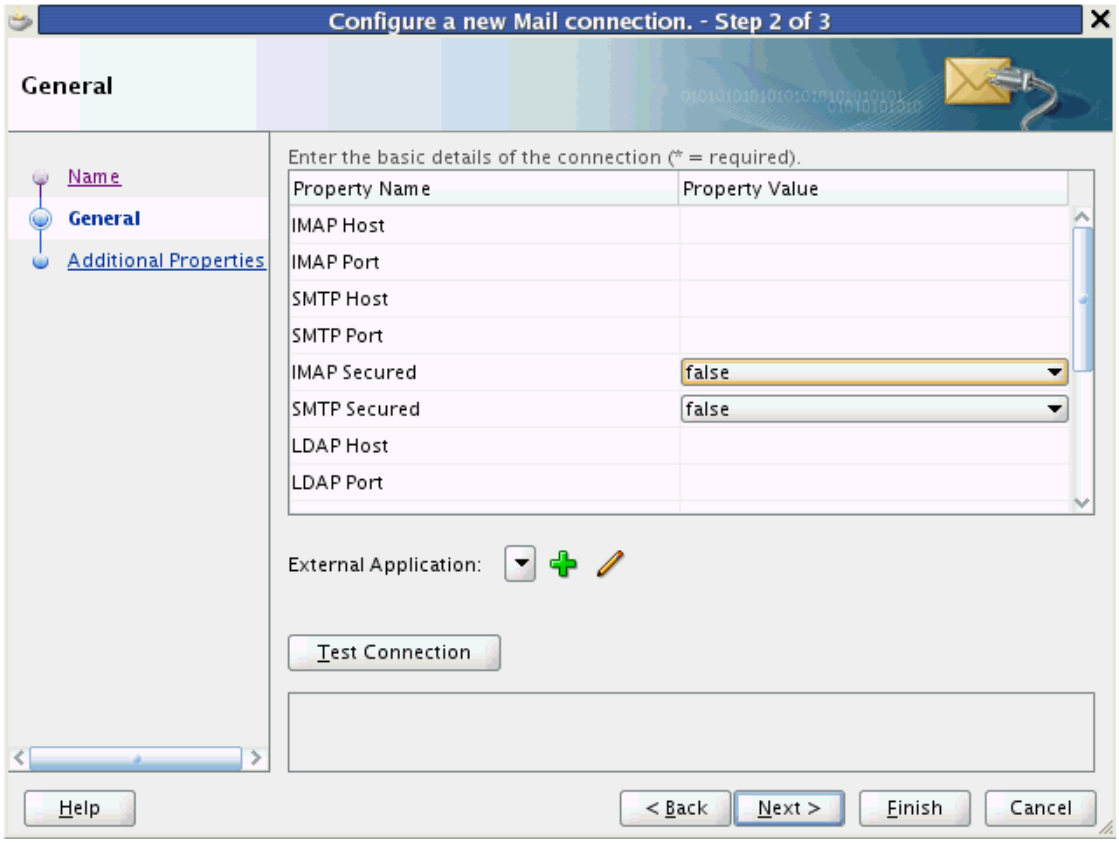
When configuring multiple mail accounts, you are not necessarily required to select this as the default connection. Keep in mind, however, that the service requires that one connection be marked as the default connection.

Note: After you create a connection as the default connection, you cannot edit it so that it is *not* the default. To use a different default connection, you must create a new connection and mark that as the default connection.

---

6. On the **General** page, enter values for the parameters (Figure 36–6).

Figure 36–6 Configure a New Mail Connection, Step 2



- `IMAP Host`: The location of your IMAP server
- `IMAP Port`: The IMAP server port number (default is -1)
- `SMTP Host`: The location of your SMTP server
- `SMTP Port`: The SMTP server port number (default is -1)
- `IMAP Secured`: Indicates (true/false) a secure SSL connection (default is false)
- `SMTP Secured`: Indicates (true/false) a secure SSL connection (default is false)
- `LDAP Host` and `LDAP Port`: These, and all other LDAP values, are required only if Microsoft Exchange is the mail server. For the Mail service to create distribution lists reliably, the WebCenter Portal: Spaces application should use the same Active Directory server as Microsoft Exchange.
- `Connection Timeout`: The connection timeout (in seconds).

7. Click **Test Connection** to check that the host and port are available.
8. You must select an existing external application or create a new external application to proceed:
   a. For **External Application**, click the **+** icon to open the Register External Application wizard

      The application maps the mail server user to the application user so that end users are not required to enter their user names and passwords each time.

For more information on external applications, see Section 69.13, "Working with External Applications."

Note: External application credential provisioning is built into the mail connection. You do not need to drop the **External Application - Change Password** task flow on a page.

---

b. On the Name page:

For **Application Name**, enter a unique name to identify the application. This name must be unique not only within the Framework application, but also among other connections. Note that you cannot edit this field afterward.

For **Display Name**, enter a name for the application that end users see in the credential provisioning screens.

c. Click **Next**.

d. On the General page, you can *optionally* enter values if you want the external application for the Mail service to participate in Click Through Login.

For **Login URL**, enter the URL to which the HTML login page is submitted. View the HTML source of the application's login form to retrieve this URL.

For **User Name/ID Field Name**, enter the label that the application uses for the user name field, for example, User Name.

For **Password Field Name** field, enter the label that the application uses for the password field, for example Password.

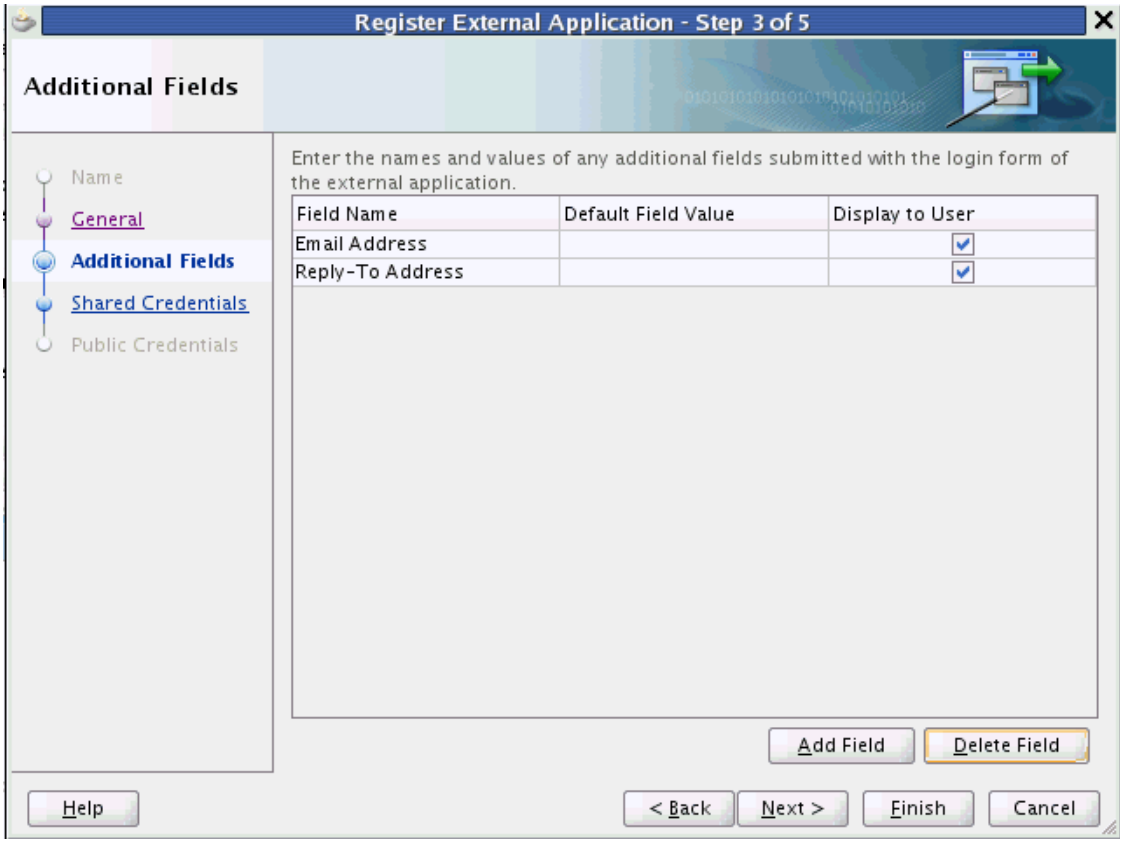
From the **Authentication Method** list, select **POST**. This submits login credentials within the body of a form. The external application for the Mail service requires this authentication method.

e. Click **Next**.

f. On the Additional Fields page, click **Add Field**, and add an extra field with the name `Email Address`. This field captures the user's mail address, so that when the user sends mail, the sender address is this mail address. Select the **Display to User** checkbox.

Additionally, to specify that replies to the user's mail should go to different mail address than the `Email Address` field, click **Add Field** again, and add an extra field with the name `Reply-To Address`. Select the **Display to User** checkbox, as shown in Figure 36–7.

Figure 36–7 Email Address and Reply-To Additional Fields



Note: The external application for the Mail service requires the `Email Address` field, and it must be shown to users. Fields for `Display Name` and `Reply-To Address` also are leveraged when sending mail.

---

g. The External Application service allows different types of credentials to be associated with a connection.

When shared credentials are specified, every *authenticated* user uses the same credentials to access the external application; that is, the user name and password you define here.

With public credentials, without authenticating your Framework application, you can view mail from a certain mail ID for all *unauthenticated* (public) users. Public credentials are used whenever an application is not secured or the user has not yet logged in.

Note: Public credentials are required to send mail from a self-registration page.

With private credentials, each user must authenticate to an *individual* mail ID. That is, each application user must specify his own credentials.

h. Click **Finish** to have the external application use private credentials, or click **Next** to set up shared or public credentials.

i. *For Shared Credentials Only*: On the Shared Credentials page, ensure that **Specify Shared Credentials** is selected and then enter the shared user credentials and mail ID.

j. *For Public Credentials Only*: On the Public Credentials page, ensure that **Specify Public Credentials** is selected and then enter the user credentials and mail ID for public use.

k. Click **Finish** to register the external application.

9. Back on the Mail connection wizard, ensure that this newly-created external application connection for mail is selected.

10. Optionally, you can add LDAP parameter values for the Active Directory server to manage (WebCenter Portal: Spaces) space distribution lists.

    For detailed parameter information, see Table 36–2.

> **Note:** For WebCenter Portal: Spaces and the Mail service to share an identity management system for setting up space mailing lists, you must use Active Directory. For information about installing and configuring mail servers as the WebCenter Portal administrator, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Table 36–2 LDAP Directory Server Configuration Parameters

Field	Description
LDAP Host	Enter the host name of the computer where the LDAP directory server is running.
LDAP Port	Enter the port on which the LDAP directory server listens.
LDAP Base DN	Enter the base distinguished name for the LDAP schema. For example, CN=Users,DC=oracle,DC=com.
LDAP Domain	Enter the domain to be appended to distribution list names. In WebCenter Portal: Spaces, for example, if the domain value is set to `oracle.com`, then the Finance Project space maintains a distribution list named `FinanceProject@oracle.com`.
LDAP Administrator User Name	Enter the user name of the LDAP directory server administrator. A valid user with privileges to make entries into the LDAP schema.
LDAP Administrator Password	Enter the password for the LDAP directory server administrator. The password is stored in a secured store.
LDAP Default User	Enter a comma-delimited list of user names to whom you want to grant moderator capabilities. These users become members of every space distribution list that is created. The users specified must exist in the base LDAP schema (specified in the `LDAP Base DN` field).
`ldap.secured`	Indicate whether a secured connection (SSL) is required between the Framework application and the LDAP directory server. If LDAP is configured to run in secure mode, then add this property (set to true/false) to use LDAP while creating distribution lists.

11. Click **Finish**.

You can see the new mail connection under **Application Resources - Connections**.

36.2.3 Adding the Mail Service at Design Time

This section explains a basic incorporation of the Mail service. It includes the following subsections:

- Section 36.2.3.1, "Mail Service Task Flows"
- Section 36.2.3.2, "How to Add the Mail Service to your Application"

36.2.3.1 Mail Service Task Flows

The Mail service includes one task flow: Mail. This task flow displays a mail inbox.

36.2.3.2 How to Add the Mail Service to your Application

To add the Mail service to your application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application.
2. Ensure that you have configured your application to connect to the mail server.
3. If you configured your external application for private or shared credentials, then the Framework application must be made secure using ADF security. ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."
4. Open the page on which to add the service.
5. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
6. Drag and drop the **Mail** task flow on to the page.
7. In the Create list, select **Region** to add the task flow to your page.
8. The resulting Edit Task Flow Binding dialog displays the optional `tabularView` parameter. If this parameter is set to `true`, then mail messages appear in a table, like an Inbox on any mail client. If this parameter is set to `false`, then mail messages render in a list view.

   Note: If you created a connection in IDE and not in the application, then the connection must be added to the application. For example, in the Resource Palette under IDE Connections, right-click the connection and select **Add to Application**.

9. Save your page and run it to the browser.
10. When you run the page with the Mail task flow for the first time, you are prompted to enter external application mail credentials from within the Mail task flow.

    Enter the credentials, then click **Submit**.

Notes:

- All instances of the Mail task flow in an application run against the same mail server, and it serves no purpose to add multiple Mail task flow instances. This is true for all service task flows that require connections to back-end servers, such as task flows from the Discussions or Announcements services.
- After an application has been accessed with saved credentials, those credentials are persisted even after redeployment. To ignore saved credentials from previous deployments, edit `appUID` in `adf-config.xml` before redeploying the application. Edit the following value in bold to specify a modified value; for example, add 1 to the last value to update it as `WCApp1-1235`).

```
 <adf:adf-properties-child
xmlns="http://xmlns.oracle.com/adf/config/properties">
     <adf-property name="adfAppUID" value="WCApp1-1234"/>
 </adf:adf-properties-child>
```

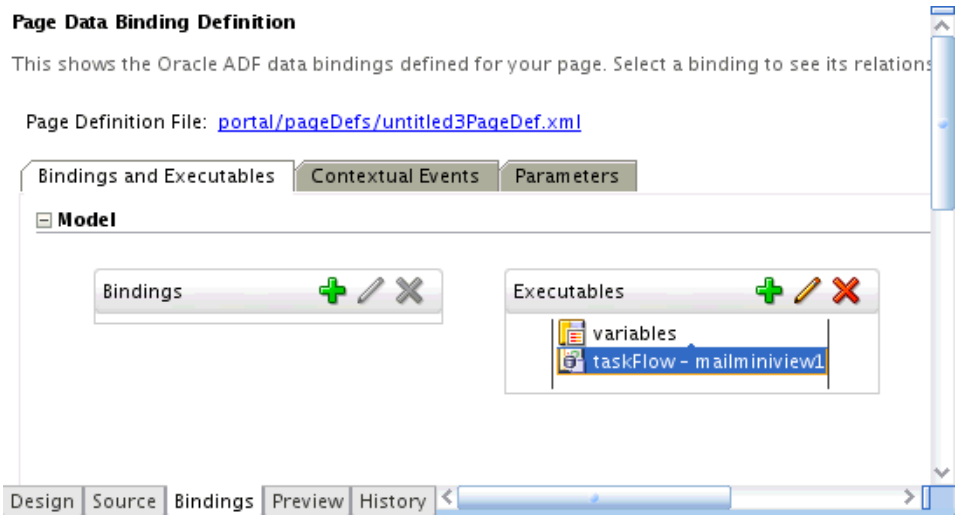
36.2.3.3 How to Modify the Mail Service Task Flow Parameters

The Mail service task flow has an optional task flow binding parameter.

You can adjust the parameter values when you drop the task flows onto a page or after you have placed a task flow on a page:

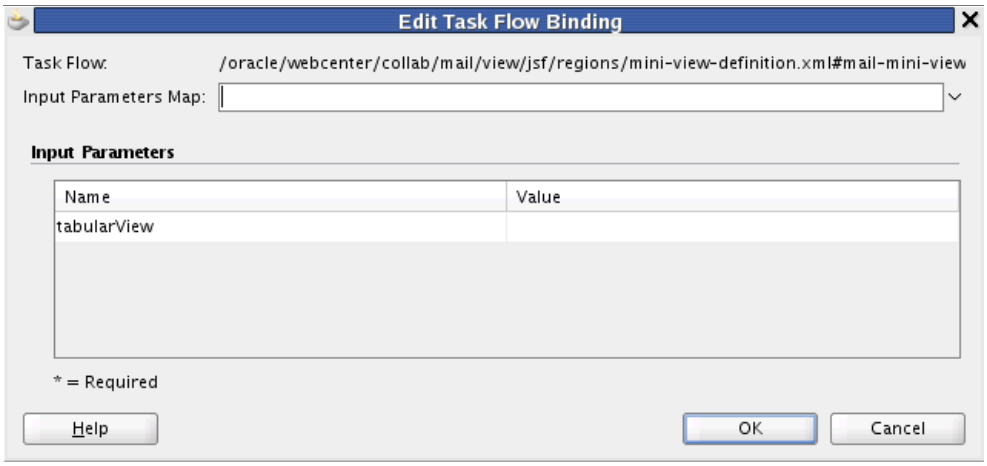
1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 36–8 shows an example of a Search task flow in the Executables section.

Figure 36–8 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog Figure 36–9, revise the binding parameter values as required.

Figure 36–9 Edit Task Flow Binding Dialog for Mail Task Flow



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

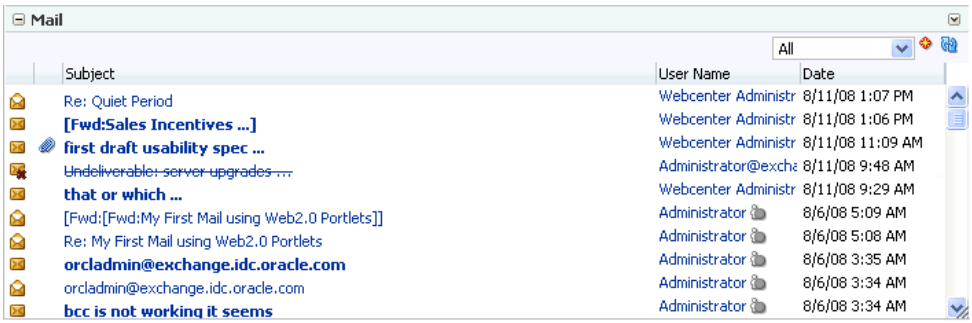
Table 36–3 describes the properties that are unique to the Mail service task flow.

Table 36–3 Mail Service Task Flow Parameters

Property	Description	Task Flow
`tabularView`	Using the EL value type, enter a value of true to display the information associated with a mail message, such as its subject, sender, and, date sent, in a tabular format. If this parameter is set to false, then mail messages render in a list view.	Mail

Figure 36–10 depicts the Mail task flow where the region parameter `tabularView` is set to true.

Figure 36–10 A Mail Task Flow where the Region Parameter Tabular Is Set to True



36.2.4 Setting Security for the Mail Service

The Mail service requires security to fetch mail for each logged-in user. ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

The external application credentials for the user name used to log in to the application are fetched and used to log in to the mail server. The recommended approach is to have the mail server and the Framework application point to the same identity store.

> **Note:** Even if the Framework application and the mail server share the same identity store, the Mail service does not support identity propagation. Single sign-on functionality is enabled through the external application mechanism.

The Mail service works in a non-secured Framework application if, and only if, the external application connection is configured with public credentials. If you do not apply security, and if mail requires a login to access the content, then users are not able to authenticate and do not see any content at runtime.

For information about using external applications, see Section 69.13, "Working with External Applications."

36.3 Advanced Information for the Mail Service

This section describes optional features available with the Mail service. It includes the following subsections:

- Section 36.3.1, "Invoking the Mail Compose Page"
- Section 36.3.2, "Configuring the Number of Mails Displayed"
- Section 36.3.3, "Troubleshooting the Mail Service"

36.3.1 Invoking the Mail Compose Page

The **Mail Compose** page enables users to determine how they want to compose individual messages within the application.

Invoke the **Mail Compose** page directly with a navigation rule that directs the user to the full page (Example 36–1).

Example 36–1 Invoking the Mail Compose Page

```
/oracle/workplace/collab/mail/view/jsf/pages/ComposeView.jspx
```

When you cannot provision mail accounts for all users within a space but want to provide the ability for space members to send mails to other members, then specify a shared (public) mail connection with the `useConnection` parameter. (Specify the connection name as a region input parameter to the mail-service compose view.) The end user Mail Preferences does not display mail connections with shared credentials.

You can pass optional parameters to seed the compose message. For example, you can pass parameters to pre-populate fields like To, Cc, Bcc, From, Subject, and Content. Set parameters only for items you want to pre-populate. If you require an empty `ComposeView.jspx`, then no `setActionListener` is necessary. You must set all parameters on `pageFlowScope`.

Use the `scope` parameter to set the scope within which the compose view should be launched.

Use the `sendMailToGSMembers` parameter to select the option to send mail to all space members.

Example 36–2 shows some parameters for the Mail Compose page (`ComposeView.jspx`):

Example 36–2 Parameters of the Mail Compose Page

```
<af:commandLink text="Compose Mail" action="sendMail">

    <af:setActionListener from="#{'john.doe@oracle.com'}"
to="#{pageFlowScope['collab.mail.compose.toList']}"/>
    <af:setActionListener from="#{'Mail......'}"
to="#{pageFlowScope['collab.mail.compose.subject']}"/>
    <af:setActionListener from="#{'Mail Service'}"
to="#{pageFlowScope['collab.mail.compose.content']}"/>
    <af:setActionListener from="#{'ruby@oracle.com'}"
to="#{pageFlowScope['collab.mail.compose.ccList']}" />
    <af:setActionListener from="#{'ruby@oracle.com'}"
to="#{pageFlowScope['collab.mail.compose.bccList']}" />
    <af:setActionListener from="#{'monty@oracle.com'}"
to="#{pageFlowScope['collab.mail.compose.from']}" />
    <af:setActionListener from="#{'text/html'}"
to="#{pageFlowScope['collab.mail.compose.contentType']}" />
    <af:setActionListener from="Shared-mail-connection-name"
to="${pageFlowScope['collab.mail.compose.useConnectionName']}" />

</af:commandLink>
```

Example 36–3 shows how to hide the recipients fields (like **To**, **Cc** and **Bcc**) by setting the following action listener in `ComposeView.jspx`:

Example 36–3 Hiding Recipient Fields in the Mail Compose Page

```
    <af:setActionListener from="#{'false'}"
to="#{pageFlowScope['collab.mail.message.showrecipients']}" />
```

36.3.2 Configuring the Number of Mails Displayed

By default, the Mail service displays the 50 most recent mail messages from your Inbox folder. Providing that your mail server supports the increase in memory cache that fetching additional mail requires, the administrator can change this to a higher number in the `adf-config.xml` file.

Add the `mail.messages.fetch.size` property as shown in Example 36–4:

Example 36–4 Increasing the Number of Mails Displayed

```
<adf-collaboration-config xmlns="http://xmlns.oracle.com/webcenter/collab/config">

<service-config
serviceId="oracle.webcenter.collab.mail">
<property name="mail.messages.fetch.size" value="500"/>
</service-config>

</adf-collaboration-config>
```

Alternatively, your Fusion Middleware administrator can increase this value with the WLST command `setMailServiceProperty`. For more information, see *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*.

This change applies to all users; that is, following Example 36–4, all users see 500 recent mails in their Inbox in the Framework application. Increasing the number of

messages shown correspondingly increases the cache size on the Framework application. Take care to set this to a reasonable size.

36.3.3 Troubleshooting the Mail Service

This section describes common problems and solutions for the Mail service.

Problem

Users are not able to retrieve their mail messages or send mail messages from within the Framework application.

Solution

Ensure the following:

- Ensure that a Mail service connection exists in your application.
- Ensure the required Mail service connection is marked as the default connection.
- Ensure the mail server configured in the connection is up and running. You can try connecting from any other mail client to ensure that the connection details are correct.

Problem

The Mail service within your Framework application requires users to log in, but users are not able to authenticate and do not see any content at runtime. When users access the Mail service, it throws the ExtApp Authorization Exception.

Solution

The Mail service works in a non-secured Framework application if, and only if, the Mail connection is configured to use an external application connection with public credentials. If your application is running in non-secured mode, then ensure that you have configured public credentials for your external application.

If you want your Framework application to run in secure mode, then you must configure ADF security for your application.

Problem

When users receive mail in Framework applications, message content is shown as an attachment (named `content.html`) rather than within the message body. This can occur if the mail server is running Microsoft Exchange Server 2007 and the "*Update Rollup 3 for Microsoft Exchange Server 2007*" is not yet installed.

Solution

Mail server administrators must download and install "*Update Rollup 3 for Microsoft Exchange Server 2007*" which fixes this issue. For more information, see `http://support.microsoft.com/kb/930468`.

37

Integrating the Polls Service

This chapter explains how to integrate the Polls service into a WebCenter Portal: Framework application at design time. For more information about using polls, see "Working with the Polls Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This chapter includes the following sections:

- Section 37.1, "Introduction to Polls"
- Section 37.2, "Basic Configuration for the Polls Service"
- Section 37.3, "Advanced Information for the Polls Service"

37.1 Introduction to Polls

The Polls service lets you create, edit, and take online polls on your application pages. Polls let you survey your audience (such as their opinions and their experience level), check whether they can recall important information, and gather feedback on the efficacy of presentations.

This section includes the following subsections:

- Section 37.1.1, "Understanding the Polls Service"
- Section 37.1.2, "Requirements for Polls"
- Section 37.1.3, "What Happens at Runtime"

37.1.1 Understanding the Polls Service

In addition to taking available polls, the Polls service lets you do the following:

- Quickly create and publish polls.
- Customize an existing poll; for example, add or change questions, and publish (open) and close the poll.
- Take a specific or the most recently-published poll
- View poll results
- See the status of all polls and available actions for each poll, such as edit, delete, publish, analyze, and clear results.

The Polls service is integrated with the Instant Messaging and Presence service in the Polls Manager.

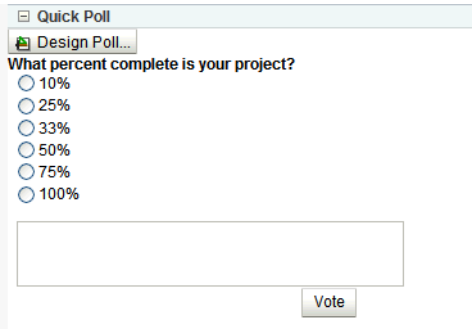
37.1.2 Requirements for Polls

In a Framework application, the Polls service requires a connection to WebCenter Portal database schema. For details about setting up a database connection, see Section 7.2.2, "Setting Up a Database Connection."

37.1.3 What Happens at Runtime

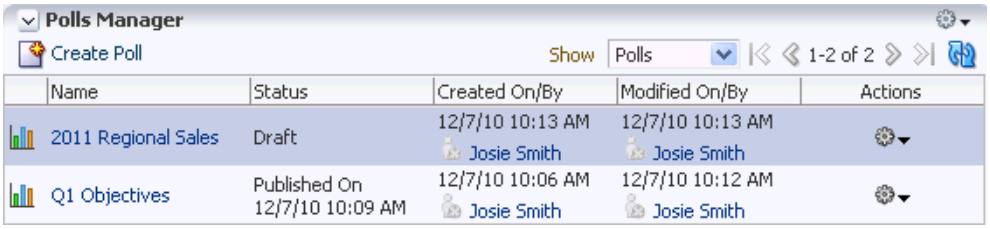
At runtime, users can quickly create polls with just a name and a question. Figure 37–1 shows a sample quick poll.

Figure 37–1 Quick Poll



WebCenter Portal application administrators can access the Polls Manager, which lets you create polls with multiple questions or with templates. The Actions dropdown list lets continue designing the poll, publish the polls, analyze results, clear results, and delete the poll (Figure 37–2).

Figure 37–2 Polls Manager



Polls created through the Polls Manager must be published and open to be taken. Users cannot take unpublished or closed polls.

For more information about the service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

37.2 Basic Configuration for the Polls Service

This section describes required steps for adding this service to your application. It includes the following subsections:

- Section 37.2.1, "Setting up Connections for the Polls Service"
- Section 37.2.2, "Adding the Polls Service at Design Time"
- Section 37.2.3, "Setting Security for Polls"

37.2.1 Setting up Connections for the Polls Service

The Polls service requires a connection to the database where the WebCenter Portal schema is installed. For details about setting up a database connection, see Section 7.2.2, "Setting Up a Database Connection."

37.2.2 Adding the Polls Service at Design Time

This section explains a basic incorporation of the Polls service. It includes the following sections:

- Section 37.2.2.1, "Polls Task Flows"
- Section 37.2.2.2, "How to Add the Quick Poll Task Flow to Your Page"
- Section 37.2.2.3, "How to Modify the Polls Service Task Flow Parameters"

37.2.2.1 Polls Task Flows

The Polls service includes the task flows described in Table 37–1.

Table 37–1 Polls Task Flows

Task Flow	Description
Poll - Quick Poll	This task flow displays a view that enables users to create one-question polls to be published immediately. Each quick poll needs its own Quick Poll task flow on the page.
Polls - Polls Manager	This task flow allows users to perform administrative operations on the polls. Any user who has access to this task flow can perform the administrative operations, such as edit or delete the poll.
Polls - Take Polls	This task flow displays the most recently-published available poll, unless it is set to a specific poll in the `pollId` parameter.
Polls - View Poll Results	This task flow displays the results of a poll as a graph for the supplied `pollId`. This task flow works only if `pollId` is supplied.

37.2.2.2 How to Add the Quick Poll Task Flow to Your Page

The Quick Poll task flow display polls and provides tools to create, edit, and delete polls. It also provides controls for determining when a poll when it expires. This task flow lets you present polls to end users where manage controls are not needed.

To add the Quick Poll task flow to your WebCenter Portal: Framework application:

1. Open the page on which you want to add the service.
2. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
3. Drag **Polls - Quick Poll** from the Resource Palette and drop it onto the page inside of the `af:form` `begin` and `end` tags.
4. When prompted, select **Region** as the way to create the task flow (and confirm with **Add Library**). This operation may take a moment to complete.
5. Click **OK**.
6. Save and run your page.

37.2.2.3 How to Modify the Polls Service Task Flow Parameters

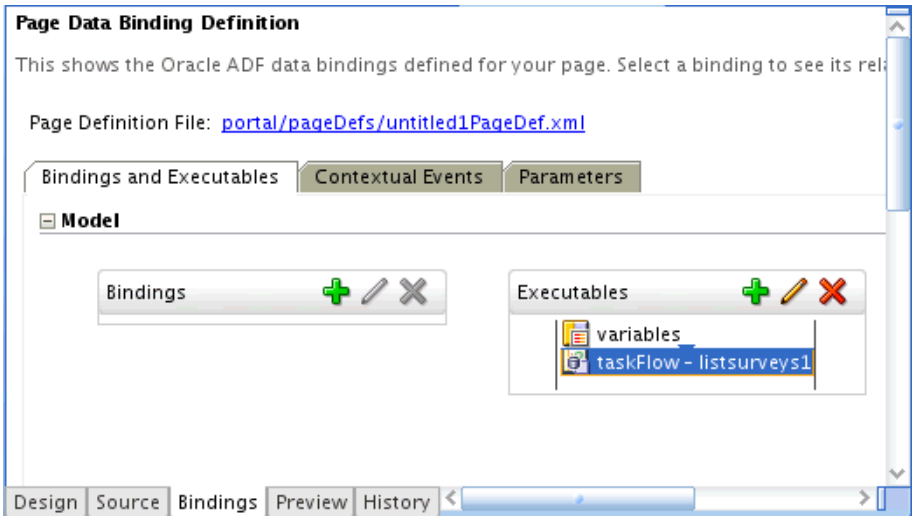
The Polls service task flows have optional task flow binding parameters.

In addition to providing required values for successful task flow rendering, you can use task flow parameters to customize the appearance and behavior of a task flow instance. For example, you can use parameter values to set which poll to display.

You can adjust the parameter values when you drop the task flows onto a page or after you have placed a task flow on a page:

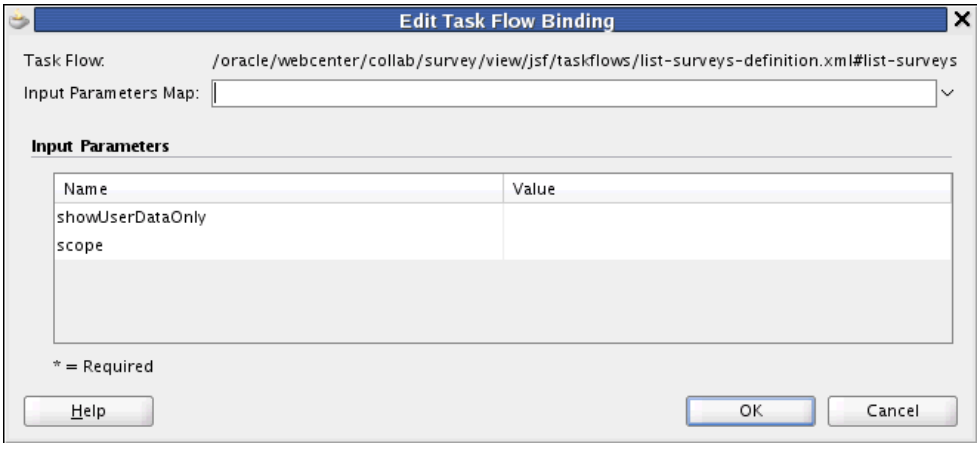
1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 37–4 shows an example of a Search task flow in the Executables section.

Figure 37–3 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog Figure 37–4, revise the binding parameter values as required.

Figure 37–4 Edit Task Flow Binding Dialog for Polls Manager Task Flow



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

Table 37–2 describes the properties that are unique to the Polls task flows.

Table 37–2 Polls Service Task Flow Parameters

Property	Description
`scope`	The space name from which polls are to be fetched. If this is supplied, then polls of that particular space are shown. On the Home space, when this parameter is not supplied, it fetches polls from all spaces. This parameter appears in the properties for the Polls Manager task flow.
`showUserDataOnly`	This parameter determines whether to display all polls or only those polls created by the user. The default (No) is to show all polls. Set to Yes to display only those polls created by the user. If the `scope` parameter is specified, then this parameter works the same but within only that scope. This parameter appears in the properties for the Polls Manager task flow.
`pollId`	The poll to display. This parameter appears in the properties for the following task flows. ■ Take Polls ■ View Poll Results
`showInEditMode`	This parameter determines whether users can edit a quick poll. When a quick poll is in Edit mode, a Design Poll box for editing the poll appears. When a quick poll is not a Edit mode, the Design Poll box for editing is not visible. This parameter appears in the properties for the Quick Poll task flow.

37.2.3 Setting Security for Polls

The Polls service does not require security. Administrators control access to Polls task flows through the Resource Catalog, where they define the users and groups privileged to see polls. Security is driven by the page security on which the task flow exists or by the task flow permissions on that page. Any user who can see the task flow can edit the poll.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

37.3 Advanced Information for the Polls Service

This section describes optional features available with this service. It includes the following subsections:

- Section 37.3.1, "How to Add the Polls Manager Task Flow"
- Section 37.3.2, "How to Add the Take Polls Task Flow"
- Section 37.3.3, "How to Add the View Poll Results Task Flow"
- Section 37.3.4, "Using the Polls Service Data Controls"

37.3.1 How to Add the Polls Manager Task Flow

The Polls Manager task flow provides a complete view of polls and perform actions on them.

To add this to your Framework application, follow the same instructions that you did for the Polls Manager task flow in Section 37.2.2, "Adding the Polls Service at Design Time," but drag and drop Polls Manager onto the page.

37.3.2 How to Add the Take Polls Task Flow

This task flow displays published polls for users to take. It displays the most recently-published available poll, unless it is set to display a specific poll with the `pollId` parameter. After a user submits a response for that poll, it displays the next most recently-published poll.

To add this to your Framework application, follow the same instructions that you did for the Polls Manager task flow in Section 37.2.2, "Adding the Polls Service at Design Time," but drag and drop Polls - Take Polls onto the page.

37.3.3 How to Add the View Poll Results Task Flow

The View Poll Results task flow displays the results of a poll as a graph for the poll supplied with the required `pollId` parameter.

To add this task flow to your Framework application, follow the same instructions that you did for the Polls Manager task flow in Section 37.2.2, "Adding the Polls Service at Design Time," but drag and drop Polls - View Poll Results onto the page.

37.3.4 Using the Polls Service Data Controls

Use the Polls service data controls to build a customized user interface for polls with a Framework application or a custom task flow.

The Polls service includes two data controls to customize the user interface:

- Take Poll Data Control
- Polls Manager Data Control

37.3.4.1 Take Poll Data Control

The Take Poll data control contains the following methods:

- createResponse
- createResponseForLatestAvailablePoll
- getPublishedPolls
- getResults
- saveResponse

37.3.4.1.1 createResponse This method lets users respond to a particular poll. Table 37–3 describes the required parameters for this method.

Table 37–3 createResponse Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.

37.3.4.1.2 createResponseForLatestAvailablePoll This method lets users respond to the most recently-published poll. It has no parameters.

37.3.4.1.3 getPublishedPolls Table 37–4 describes the required parameters for this method.

Table 37–4 getPublishedPolls Input Parameters

Parameter	Type	Description
`startIndex`	Integer	The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.
`numResults`	Integer	The number or results to display.

37.3.4.1.4 getResults This method gets the poll results for a particular poll. Table 37–5 describes the required parameters for this method.

Table 37–5 getResults Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.

37.3.4.1.5 saveResponse Table 37–6 describes the required parameters for this method.

Table 37–6 saveResponse Input Parameters

Parameter	Type	Description
`response`	Response	The response object acquired from the data control calls to `createResponse` or `createResponseForLatestAvailablePoll`.

37.3.4.2 Polls Manager Data Control

The Polls Manager data control contains the following methods:

- addQuestion
- addQuestion
- closePoll
- closePollAfterDate
- closePollAfterDays
- closePollAfterResponses
- createPoll
- deletePoll
- editPoll
- editPoll
- findPoll
- getPolls
- getPollsByUser
- getPollsByUserCount
- publishPoll
- publishPollOnDate

37.3.4.2.1 addQuestion This method creates polls with multiple choice questions. Table 37–7 describes the required parameters for this method.

Table 37–7 addQuestion Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`question`	String	The text of the question; for example, What is your favorite color?
`rowOptions`	String	Corresponding list of options for the question. This should be line-separated list of options.
`allowMultipleSelection`	Boolean	True to allow multiple options to be selected; false to allow only a single option selection.

37.3.4.2.2 addQuestion This method creates complex polls with matrix-type questions. Table 37–8 describes the required parameters for this method.

Table 37–8 addQuestion Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`questionType`	Question Type	Question type, either: ■ Matrix of choices (only one answer per row) ■ Matrix of choices (multiple answers per row)
`question`	String	The text of the question; for example, What is your favorite color?
`rowOptions`	String	Corresponding list of options for the question. This should be line-separated list of options.
`columnOptions`	String	Column options.
`isOptional`	Boolean	True if the question is optional (that is, users can ignore answering it); false if question is mandatory.
`commentLabel`	String	Enables a comment field for users to provide feedback or enter personalized text with their answers.
`commentFieldHeight`	Integer	Size for the comment field. To capture feedback, set this to 3; otherwise, set it to 1. This parameter value depends on the type of answer expected in the comment field.

37.3.4.2.3 closePoll This method closes a particular poll. Table 37–9 describes the required parameters for this method.

Table 37–9 closePoll Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.

37.3.4.2.4 closePollAfterDate This method closes a poll on a specific date. Table 37–10 describes the required parameters for this method.

Table 37–10 closePollAfterDate Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`date`	Date	Date at which the poll should close.

37.3.4.2.5 closePollAfterDays This poll closes a poll at a set number of days after the poll is published. Table 37–11 describes the required parameters for this method.

Table 37–11 closePollAfterDays Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`days`	Integer	Number of days, after the publish date, at which the poll should close.

37.3.4.2.6 closePollAfterResponses This method closes as poll after a set number of responses have been collected. Table 37–12 describes the required parameters for this method.

Table 37–12 closePollAfterResponses Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`responses`	Integer	Number of responses at which the poll should close.

37.3.4.2.7 createPoll This method creates a quick poll. Table 37–13 describes the required parameters for this method.

Table 37–13 createPoll Input Parameters

Parameter	Type	Description
`name`	String	Name of the poll.
`description`	String	Description of the poll.

37.3.4.2.8 deletePoll This method deletes a particular poll. Table 37–14 describes the required parameters for this method.

Table 37–14 deletePoll Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.

37.3.4.2.9 editPoll Table 37–15 describes the required parameters for this method.

Table 37–15 editPoll Input Parameters

Parameter	Type	Description
`pollInstance`	Poll	If a poll instance is directly edited, then that instance can be supplied for editing.

37.3.4.2.10 editPoll This method edits a particular poll. Table 37–16 describes the required parameters for this method.

Table 37–16 editPoll Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`name`	String	Name of the poll.
`description`	String	Description of the poll.

37.3.4.2.11 findPoll This method finds a particular poll. Table 37–17 describes the required parameters for this method.

Table 37–17 findPoll Input Parameters

Parameter	Type	Description
`id`	String	Poll instance ID

37.3.4.2.12 getPolls This method displays all polls, as defined in its parameters. Table 37–18 describes the required parameters for this method.

Table 37–18 getPolls Input Parameters

Parameter	Type	Description
`startIndex`	Integer	The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.
`numResults`	Integer	The number or results to display.

37.3.4.2.13 getPollsByUser This method displays polls created by a particular user. Table 37–19 describes the required parameters for this method.

Table 37–19 getPollsByUser Input Parameters

Parameter	Type	Description
`createdBy`	String	User name of the person who created the poll.
`startIndex`	Integer	The index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.
`numResults`	Integer	The number or results to display.

37.3.4.2.14 getPollsByUserCount Table 37–20 describes the required parameters for this method.

Table 37–20 getPollsByUserCount Input Parameters

Parameter	Type	Description
`createdBy`	String	User name of the person who created the poll.

37.3.4.2.15 publishPoll This method publishes a particular poll. Table 37–21 describes the required parameters for this method.

Table 37–21 publishPoll Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.

37.3.4.2.16 publishPollOnDate This method publishes a poll on a specific date. Table 37–22 describes the required parameters for this method.

Table 37–22 publishPollOnDate Input Parameters

Parameter	Type	Description
`pollId`	String	The poll to display.
`date`	Date	Date at which the poll should be published.

Part VI

Working with the People Connections Service

This part provides comprehensive information about the People Connections service, including what it delivers and how to use it in aWebCenter portal application.

This part includes the following chapters:

- Chapter 38, "Introducing the People Connections Service"
- Chapter 39, "Basic Configuration for the People Connections Service"
- Chapter 40, "People Connections Task Flow Binding Parameters"
- Chapter 41, "Using People Connections Data Controls and Java APIs"
- Chapter 42, "People Connections Service REST APIs"

38

Introducing the People Connections Service

The People Connections service provides social networking tools for use in enhancing connection and communication within a project team and throughout an enterprise. It does this through a set of features that include:

- **Activity Stream** for viewing user activities generated through application or social networking actions.
- **Connections** for connecting to other application users to share information, comment on performance, exchange messages, and track activity
- **Feedback** for giving *ad hoc* performance feedback to other users
- **Message Board** for posting messages to other users
- **Profile** for entering personal contact information and viewing the contact information of other users
- **Publisher** for publishing status messages and posting files and links

This chapter describes the task flows associated with the People Connections service and provides some troubleshooting tips.

This chapter includes the following sections:

- Section 38.1, "Overview of the People Connections Service"
- Section 38.2, "Troubleshooting the People Connections Service"

38.1 Overview of the People Connections Service

The People Connections service provides social networking tools for creating, interacting with, and tracking the activities of one's connections. Its features enable users to manage their personal profiles, access the profiles of other users, provide *ad hoc* feedback, post messages, track activities, and connect with others.

In a production environment, an enterprise can leverage its back-end identity store as a means of providing People Connections with a population of potential connections. In a development environment, developers can add test-users to the `jazn-data.xml` file.

> **See Also:** For information about connecting to a back-end (LDAP) identity store for the production version of your application, see the chapter, "Managing Security," in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. For information about creating test users in `jazn-data.xml`, see Section 24.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File."

This section provides an overview of the People Connections service. It includes the following subsections:

- Section 38.1.1, "People Connections Service Task Flows"
- Section 38.1.2, "People Connections Service Requirements"

38.1.1 People Connections Service Task Flows

The features of the People Connections service fall into five categories. Each category includes a set of task flows that expose People Connections features to your end users. Note that the Publisher task flow does not belong to these categories, but rather, works in concert with the Activity Stream.

This section introduces these categories. It includes the following subsections:

- Section 38.1.1.1, "Activity Stream"
- Section 38.1.1.3, "Feedback"
- Section 38.1.1.4, "Message Board"
- Section 38.1.1.2, "Connections"
- Section 38.1.1.5, "Profile"
- Section 38.1.1.6, "The Publisher Task Flow"

> **See Also:** For information about the People Connections service at runtime, see Part VIII, "Organizing Your Collaborative and Social Networking Environment," in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
>
> For information about People Connections task flows, see Section 38.1.1, "People Connections Service Task Flows."

38.1.1.1 Activity Stream

The Activity Stream feature tracks the application activities of a user's connections. Table 38–1 lists the types of activities that may be tracked by the Activity Stream.

Table 38–1 Activities Tracked by Activity Stream

Service	Tracked Activities
Announcements	■ Create announcement ■ Edit announcement
Blogs	■ Create blog ■ Update blog
Connections	■ Invitations to connect ■ People are connected
Discussions	■ Create forum ■ Create topic ■ Reply to topic
Documents	■ Create document ■ Edit document ■ Add tag ■ Remove tag

Table 38–1 (Cont.) Activities Tracked by Activity Stream

Service	Tracked Activities
Events	■ Create an event ■ Edit an event
Feedback	■ Feedback left ■ Feedback received
Lists	■ Create a list ■ Add a row to a list ■ Edit a list row
Message Board	■ Message left ■ Message received
Pages	The following activities are tracked when they are performed within the service framework scope (for example, if a user creates pages under `/myapp/mypages/`, the page operations are not tracked): ■ Create page ■ Edit page ■ Delete page ■ Add tag
Profiles	■ Photo updated ■ Profile updated ■ Personal status note updated
Tagging	■ Add tag ■ Remove tag

Activity Stream compares somewhat to the Recent Activities service, which also tracks and reports on application activities (for more information, see Chapter 50, "Integrating the Recent Activities Service"). Both track the application activities of integrated services, though Activity Stream tracks a broader range of services. For example, Recent Activities tracks the Documents (including wikis and blogs), Announcements, Discussions, and Page services. Activity Stream tracks these services as well as People Connections. Recent Activities tracks activities no matter who performs the action. Activity Stream tracks activities performed by a user's connections and includes information about who performed the activity. Recent Activities does not include names.

The basic difference between these two services can be summarized as follows: Recent Activities provides an overview of what is happening in an application. Activity Stream provides an overview of what is happening with a user's connections.

The Publisher task flow works in concert with Activity Stream to provide content in the form of user messages. A message posted through the Publisher task flow is published to the Activity Streams of the user who entered it and that user's connections.

Activity Stream messages may include attachments, such as a file or a link. Depending on a file's mime type, Activity Stream allows for displaying a preview of such attachments. Activity Stream previews files through either a native web format or through UCM slide rendition. The previewer used depends on the mime type of the file to be previewed.

The mime types that use the native web format include:

- `image`
- `htm`
- `text`

The mime types that use UCM slide rendition include:

- `pdf`
- `powerpoint`
- `powerpnt`
- `pptx`

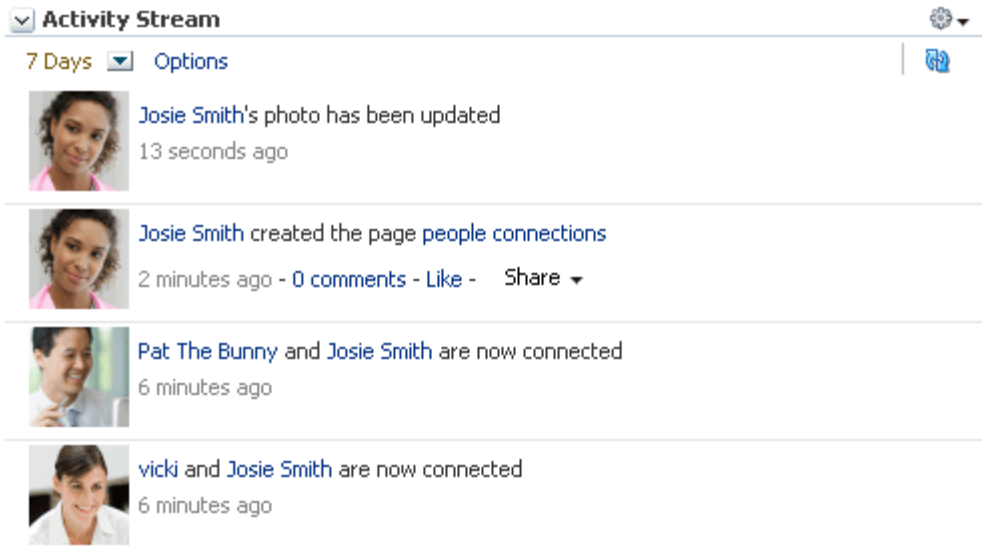
> **Note:** PDF file previews are available in Activity Stream when the mime type is `pdf`, `webContextRoot` is specified in the UCM connection, and the WebCenter Portal: Spaces application is accessed through an Oracle HTTP Server.

Note that the mime types shown in these bullet lists are the only mime types that are previewed. Other mime types appear as links. The mime types `docx` and `xlsx` are *not* previewed in Activity Stream. Note also that a parameter associated with an Activity Stream task flow instance can be set to omit file previews (for more information, see Section 40.2, "People Connections Task Flow Binding Parameters").

Two Activity Stream task flows are available through the Resource Palette:

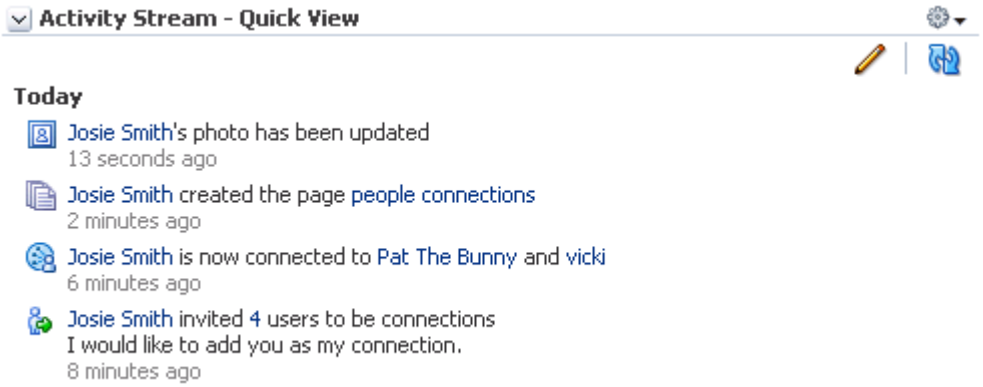
- Activity Stream—Provides users with the main view of the most recent activities of their connections (Figure 38–1).

Figure 38–1 Activity Stream Task Flow



- Activity Stream - Quick View—Provides users with a truncated view of streamed activities, where only the most recent of their connections' activities are shown. Users who want to see all activities can click a **More** link to navigate to the main view (Activity Stream) of the task flow (Figure 38–2).

Figure 38–2 Activity Stream - Quick View Task Flow



See Also: For information about Activity Stream task flows at runtime, see the section, "Working with Activity Stream Task Flows," in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

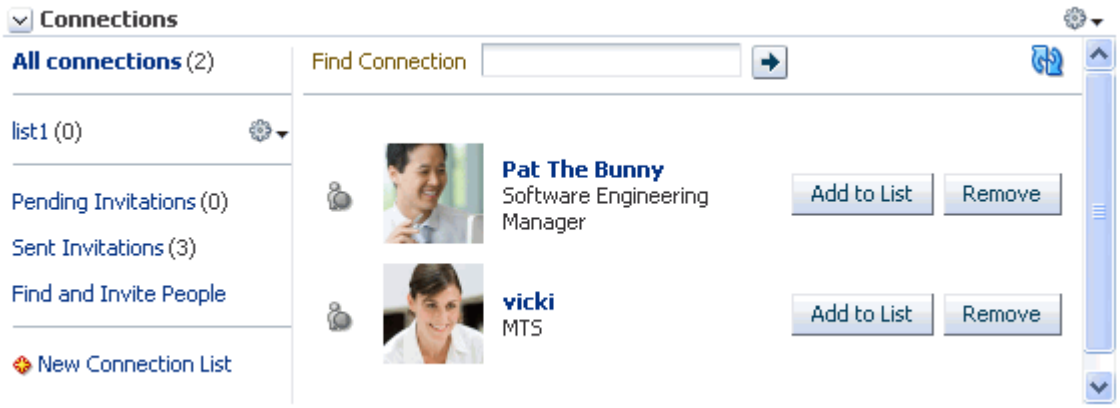
38.1.1.2 Connections

Connections provides users with a means of managing their own connections and viewing the connections of others. Use Connections to collect business friends and contacts into one or more smaller social groups. Use connections lists to manage the display of your connections.

Three Connections task flows are available through the Resource Palette:

- Connections—Provides users with a means of viewing and managing their connections, creating connections lists, and sending and responding to invitations to connect (Figure 38–3).

Figure 38–3 Connections Task Flow



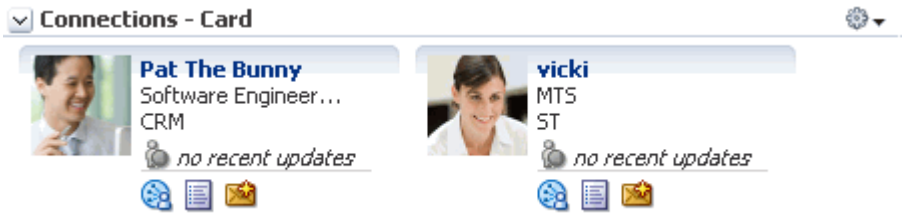
- Connections - Card—Provides a choice of the following views:
  - `vcard` displays a connection's photo, job title, and status message.
  - `iconic` displays a connection's photo and user name.
  - `list` displays connections in a list, with each list row showing the profile photo, user name, job title, and an **Add to List** button.

– `tiled` renders the connection's personal Profile photo and shows the user name and job title beside the photo.

Tip: In all views, the name links to a summary view of the user's Profile.

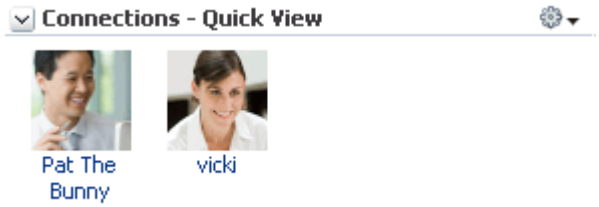
Select one of these views through task flow bindings (for more information, see Chapter 40, "People Connections Task Flow Binding Parameters") (Figure 38–4).

Figure 38–4 Connections - Card Task Flow



- Connections - Quick View—Provides users with a view of their connections' uploaded profile photos and a link to a summary view of their Profiles (Figure 38–5).

Figure 38–5 Connections - Quick View Task Flow



See Also: For information about Connections task flows at runtime, see the section, "Setting Up Your Social Network," in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

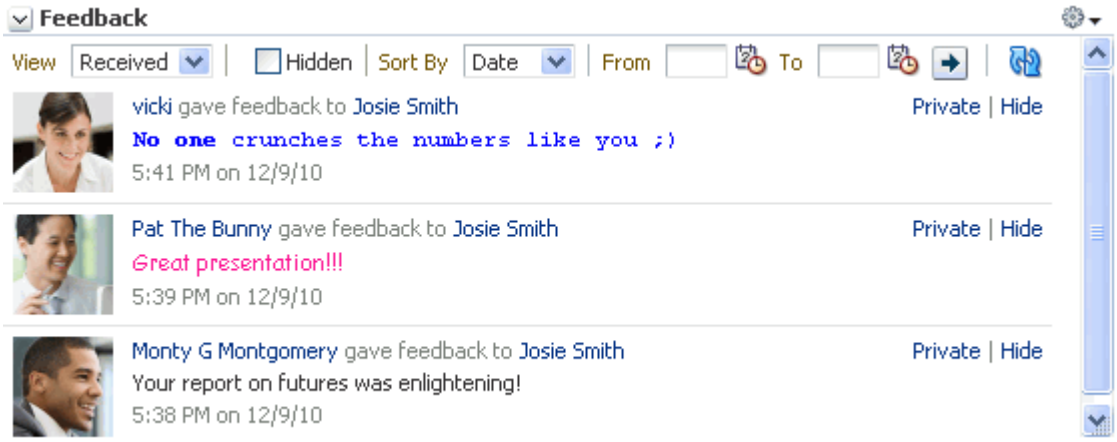
38.1.1.3 Feedback

Feedback provides users with a means of viewing, posting, and managing feedback. By default, users can view feedback in their own Feedback views. Users can view and post feedback in other users' Feedback views.

Two Feedback task flows are available through the Resource Palette:

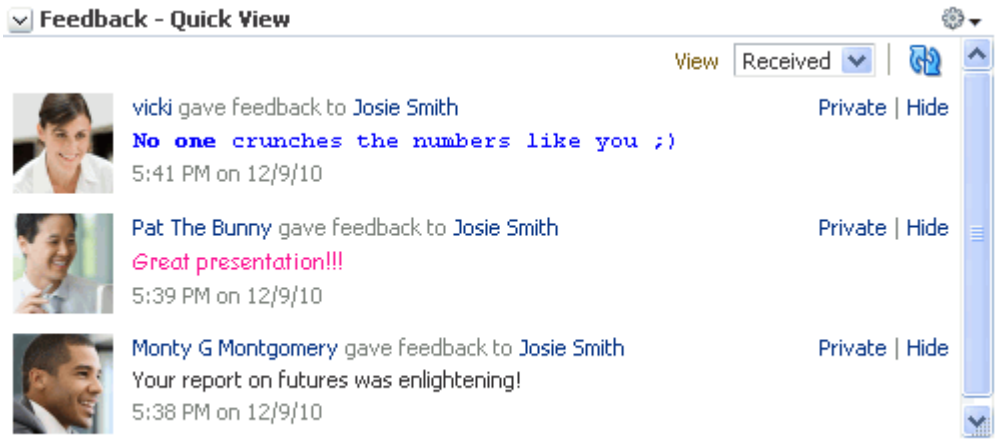
- Feedback—Provides users with the main view of the Feedback task flow. The main view contains controls for viewing, posting, sorting, and filtering Feedback—both given and received—and for hiding received Feedback (Figure 38–6).

Figure 38–6 Feedback Task Flow



- Feedback - Quick View—Provides users with a truncated view of the Feedback task flow, where only the most recent posts are shown. When there are more posts than can be displayed in the initial view, users can click a **More** link to navigate to the main view of the Feedback task flow. When user `A` accesses user `B`'s view of Feedback - Quick View, user `A` additionally sees an **Add Feedback** option (Figure 38–7).

Figure 38–7 Feedback - Quick View Task Flow



See Also: For information about Feedback task flows at runtime, see the section, "Working with Feedback Task Flows," in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

38.1.1.4 Message Board

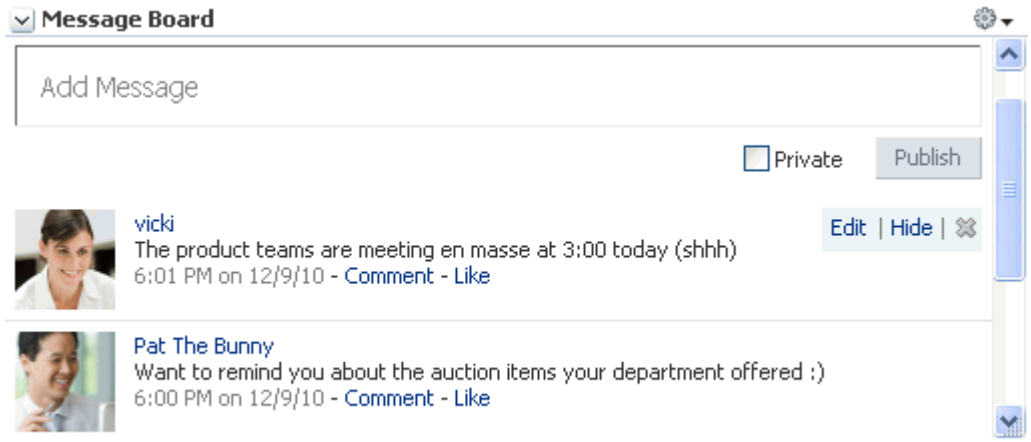
Message Board provides a means of viewing and posting messages and attachments to Message Boards and Activity Streams.

Two Message Board task flows are available through the Resource Palette:

- Message Board—Message Board provides users with the main view of Message Board messages and a means of adding, viewing, updating, hiding, deleting, and

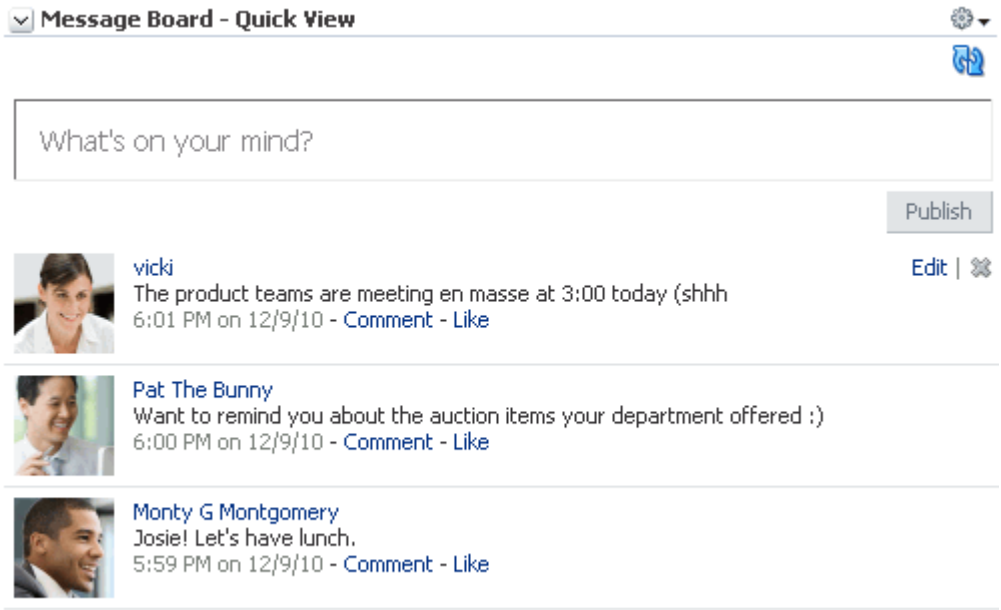
managing your view of messages, and for marking messages as private and sharing private messages (Figure 38–8).

Figure 38–8 Message Board Task Flow



- Message Board - Quick View—Provides users with a truncated view of Message Board messages and a means of adding, viewing, updating, hiding, and deleting messages, and for marking messages as private and sharing private messages. Users can click a **More** link to navigate to the Detailed View (Figure 38–9).

Figure 38–9 Message Board - Quick View Task Flow



See Also: For information about Message Board task flows at runtime, see the section, "Working with Message Board Task Flows" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

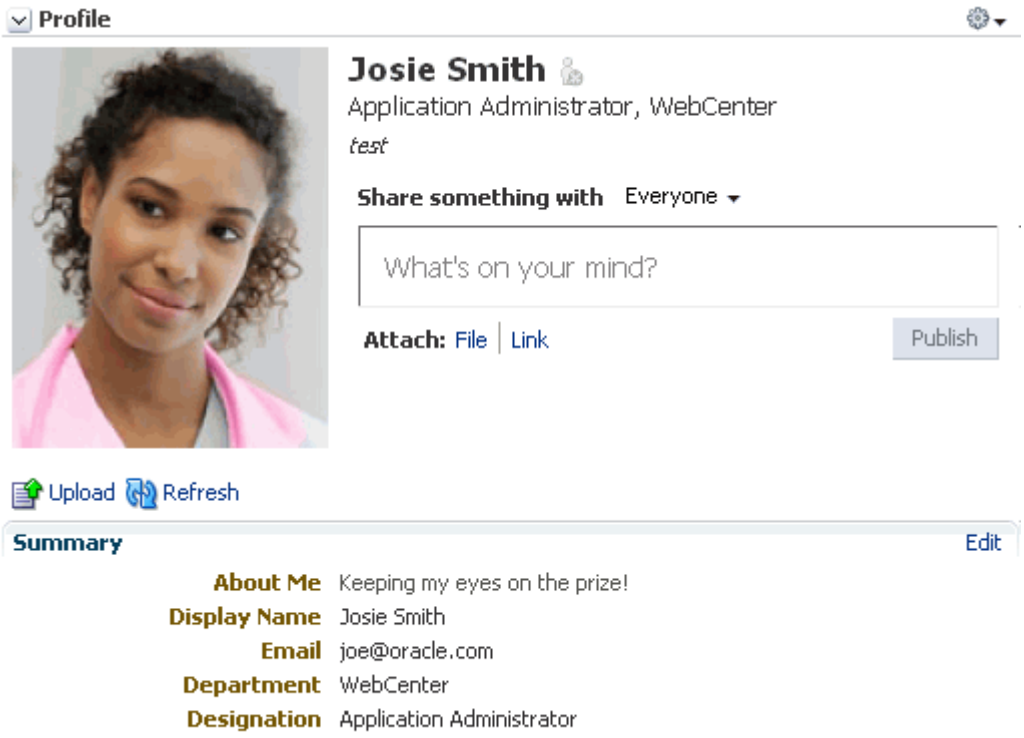
38.1.1.5 Profile

Profile provides users with a variety of views into their own and other users' personal profile information. Such information can include a user's email address, phone number, office location, department, manager, direct reports, and so on. Profile takes the bulk of its information from the back-end identity store that provides your WebCenter Portal application with its users. Additionally, Profile may offer opportunities for altering some of this information and for providing additional data not included in the identity store.

Three Profile task flow are available through the Resource Palette:

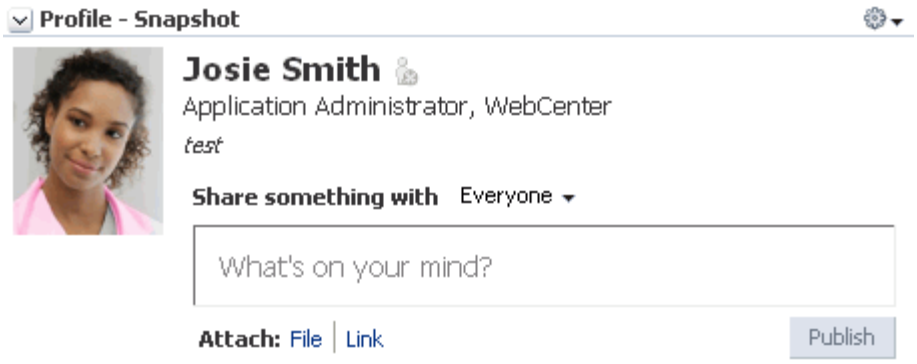
- Profile—Exposes a user's profile details, such as a user's email address, phone number, office location, department, manager, direct reports, and so on. It also exposes any personal image a user may have provided and the user's personal status message (Figure 38–10).

Figure 38–10 Profile Task Flow



- Profile - Snapshot—Exposes any personal image a user may have uploaded and provides a means of publishing messages, files, and links to a selected audience (Figure 38–11).

Figure 38–11 Profile - Snapshot Task Flow



Note: To enable users to upload anything to your ADF-based application at runtime, the page on which you have placed the upload task flow must have the `usesUpload` attribute set on the form. Additionally, for profile images, you must set the task flow binding parameter `photoUploadAllowed` to `true`. For more information, see the Note at the end of Section 39.2, "How to Add People Connections Task Flows to a Page," and see Chapter 40, "People Connections Task Flow Binding Parameters."

- Profile Gallery—Provides a summary view of each People Connections task flow. For example, it includes Activity Stream - Quick View, Connections - Quick View, and so on (Figure 38–12).

Figure 38–12 Profile Gallery Task Flow



See Also: For information about Profile task flows at runtime, see the section, "Viewing and Editing Personal Profiles" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

38.1.1.6 The Publisher Task Flow

The Publisher task flow works in concert with Activity Stream. Users enter content into Publisher, which is then published to the Activity Stream of the user who posted it as well as to the Activity Streams of that user's connections (Figure 38–13).

Figure 38–13 Publisher Task Flow



See Also: For information about the Publisher task flow at runtime, see the section, "Sharing Items, Files, and URLs Through the Publisher Task Flow" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

38.1.2 People Connections Service Requirements

For a successful integration of People Connections with your WebCenter Portal: Framework application, ensure that the following steps have been taken:

1. Create a WebCenter Portal: Framework application (for more information, see Section 5.2, "Creating a Framework Application").
2. Set up the WebCenter schema, and create a database connection to it (for more information, see Chapter 7, "Preparing Your Application for WebCenter Portal Services").
3. Create a JSF (`.jspx`) page (for more information, see Section 5.3, "Adding Pages to a Portal").
4. Secure your application and create some test users (for more information, see Chapter 69, "Securing Your WebCenter Portal: Framework Application," and Section 24.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File").
5. Drag and drop a People Connections task flow onto your page (for more information, see Section 39.2, "How to Add People Connections Task Flows to a Page").
6. Provide required values for task flow bindings (for more information, see Chapter 40, "People Connections Task Flow Binding Parameters").
7. Run the page.
8. When required, provide the user name and password of a test user you created at Step 4.

38.2 Troubleshooting the People Connections Service

This section provides information to assist you in troubleshooting problems you may encounter while using the People Connections service.

People Connections task flows take up the full browser, even when empty

This problem may arise if you are using Internet Explorer, version 8 (IE8). In IE8, go to **Tools**, **Compatibility View Settings**. In the Compatibility View Settings dialog, clear the following check boxes:

- Display intranet sites in Compatibility View
- Display all websites in Compatibility View

New Profile Image is Not Shown

Changes to a Profile image may not be rendered immediately due to a stale WebCenter Database cache. Clicking the **Refresh** icon under the stale image will cause the image to update.

39

Basic Configuration for the People Connections Service

This chapter provides the required steps for adding the People Connections service to your WebCenter Portal: Framework application. It includes the following sections:

- Section 39.1, "How to Set Up a Database Connection for the People Connections Service"
- Section 39.2, "How to Add People Connections Task Flows to a Page"
- Section 39.3, "Setting Security for the People Connections Service"
- Section 39.4, "Establishing Site-Level Settings for People Connections Features"

39.1 How to Set Up a Database Connection for the People Connections Service

For information about setting up a database connection in support of Oracle WebCenter Portal: Services, see Section 7.2.2, "Setting Up a Database Connection."

39.2 How to Add People Connections Task Flows to a Page

This section describes how to add a People Connections task flow to an application page. The steps provided here are largely the same for all People Connections task flows. Differences are noted.

To add People Connections task flows to your WebCenter Portal: Framework application:

1. Prepare your application as described in Section 38.1.2, "People Connections Service Requirements."
2. Open the page on which to add a People Connections task flow.
3. In the Resource Palette, open **My Catalogs**, then **WebCenter Portal - Services Catalog**, then the **Task Flows** folder.
4. Drag and drop a task flow (for example, Connections - Quick View) onto the JSF (`.jspx`) page.
5. Select **Region** from the resulting context menu.

   It may ask whether you want to add the People Connections library to the project. Confirm by clicking **Add Library**.
6. In the Edit Task Flow Binding dialog, specify parameter values for the task flow.

For example, for Connections - Quick View, the parameter `userid` represents the name of the current user at runtime. You can specify an EL expression for the parameter that will evaluate to the currently logged-in user. For example, for `userid`, enter `#{securityContext.userName}`. For more information, see Chapter 40, "People Connections Task Flow Binding Parameters."

7. Click **OK**.

   The task flow is added to the page, and the ViewController project's libraries are configured to run the task flow.

8. Save and run your page to the browser.

Note: To enable users to upload anything to your ADF-based application at runtime, the page on which you have placed the task flow must have the `usesUpload` attribute set on the form. For example, to enable the upload of a document to the document library or a photo to a profile, add the following to the relevant page:

```
<af:form usesUpload="true">
  …
  <af:region value="#{bindings.profile1.regionModel}"/>
  …
</af:form>
```

Additionally, to enable the upload of a profile snapshot, the task flow binding parameter `photoUploadAllowed` must be set to `true`. For more information, see Chapter 40, "People Connections Task Flow Binding Parameters."

39.3 Setting Security for the People Connections Service

Because People Connections features are centered around users, application security must be set up for successful use of the service. Ideally, test users are also in place to enable you to interact in a meaningful way with each feature. Each feature in the People Connections service can be secured separately.

- ADF security is configured by default when you create your application using the WebCenter Portal Application template. For details about how to implement a basic security solution for your WebCenter Portal application, see Section 69.3, "Configuring ADF Security."
- To create test users, follow the steps outlined in Section 24.2.2, "How to Define Roles and Grant Privileges in the jazn-data.xml File."

  Once you have added test users, you can add additional user attributes, such as `business_email`, `title`, or `department`:

  1. Open the `jazn-data.xml` file.

     Tip: You will find the `jazn-data.xml` file in the Application Resources panel in the META-INF folder under Descriptors.

  2. Select the tab for **Source** at the bottom of the main content area.
  3. Inside the `<user>` node for a selected user, insert the following:

     ```
     <property name="[property-name]" value="[property-value]"/>
     ```

For example, to add `business_email` as `monty@domain.com` for the user `monty`, add:

```
<user>
     <name>monty</name>
     <display-name>monty</display-name>
     <credentials>{903}EfEepsY6I/TrKthpcmjJIsrlP36Ysr/j</credentials>
     <property name="business_email" value="monty@domain.com"/>
</user>
```

39.4 Establishing Site-Level Settings for People Connections Features

Two scripts are available to configure and revise People Connections site-level settings:

RCUHOME/rcu/integration/webcenter/sql/oracle/settings-insert.sql

RCUHOME/rcu/integration/webcenter/sql/oracle/settings-update.sql

The variable *RCUHOME* refers to your install location of the Resource Creation Utility (RCU). The RCU may be used in setting up the WebCenter schema and is packaged with the Oracle JDeveloper ship home. `RCUHOME` is the root folder where the RCU is installed.

This section describes the scripts and provides information about the types of settings they control. It includes the following subsections:

- Section 39.4.1, "Understanding the People Connections Site-Level Setting Scripts"
- Section 39.4.2, "Supported Site-Level Settings for People Connections Features"
- Section 39.4.3, "Example: Configuring Connections to Accept Invitations Automatically"

39.4.1 Understanding the People Connections Site-Level Setting Scripts

All site-level settings for People Connections features are stored in the WebCenter schema table `WC_PPL_COMMON_SETTING`. Out-of-the-box, this table does not contain any setting values. In the absence of values, the application assumes default values. To change the value of any setting, the setting must first be inserted into this table and then updated with the desired value. Two SQL scripts enable you to perform these steps in their proper sequence.

This section describes how to prepare and run the People Connections site-level settings scripts. It includes the following subsections:

- Section 39.4.1.1, "Preparing and Running settings-insert.sql"
- Section 39.4.1.2, "Preparing and Running settings-update.sql"

39.4.1.1 Preparing and Running settings-insert.sql

Before you can change People Connections site-level settings, you must run the `settings-insert.sql` script once. The script has `INSERT` statements for all supported settings. All `INSERT` statements are commented out by default. Before you run the script, you must prepare it by uncommenting all the settings you plan to change.

> **See Also:** For an example of the `settings-insert.sql` script in action, see Section 39.4.3, "Example: Configuring Connections to Accept Invitations Automatically."

To prepare and run the `settings-insert.sql` script:

1. Open the script in an editor.
2. For the settings you plan to change, uncomment the corresponding `INSERT` statement.

   Tip: To uncomment an `INSERT` statement, remove the leading double dash (--) on the lines the statement spans.

3. Once you have uncommented all the `INSERT` statements for the settings of interest, save and run the script.

   The uncommented settings are populated with default values.

   Note: You can run the `settings-insert.sql` script only once for a given set of settings on a given schema. If you must run it more than once—for example, if you must update a different set of settings—then the previously uncommented `INSERT` statements must be commented before you can run the script again. Otherwise, a SQL error is thrown during the re-execution of the `INSERT` statement.

39.4.1.2 Preparing and Running settings-update.sql

To change People Connections site-level settings, you must run the `settings-update.sql` script after you run `settings-insert.sql` once. The `settings-update.sql` script has `UPDATE` statements for all supported settings. All `UPDATE` statements are commented out by default.

You can run the `settings-update.sql` script as many times as required for a given set of settings on a given schema, provided you have run `settings-insert.sql` once for those settings.

See Also: For an example of the `settings-update.sql` script in action, see Section 39.4.3, "Example: Configuring Connections to Accept Invitations Automatically."

To prepare and run the `settings-update.sql` script:

1. Open the script in an editor.
2. For the settings you plan to change, uncomment the corresponding `UPDATE` statement.

   Tip: To uncomment an `INSERT` statement, remove the leading double dash (--) on the lines the statement spans.

3. For each uncommented statement, change the value of the `SETTING_KEY` column to the desired value.

   For information about settings and values, see Section 39.4.2, "Supported Site-Level Settings for People Connections Features."

4. Once you have revised all `UPDATE` statements of interest, save and run the script.

   The table `WC_PPL_COMMON_SETTING` is updated with the revised values.

5. If your WebCenter Portal: Framework application is running, restart it for the changes to take effect.

39.4.2 Supported Site-Level Settings for People Connections Features

This section lists and describes the supported site-level application settings for the People Connections service. It includes the following subsections:

- Section 39.4.2.1, "Activity Stream Site-Level Settings"
- Section 39.4.2.2, "Connections Site-Level Settings"
- Section 39.4.2.3, "Feedback Site-Level Settings"
- Section 39.4.2.4, "Message Board Site-Level Settings"
- Section 39.4.2.5, "Profile Site-Level Settings"

39.4.2.1 Activity Stream Site-Level Settings

Table 39–1 lists and describes the site-level settings for the People Connections service Activity Stream feature.

The service ID for Activity Stream is `oracle.webcenter.activitystreaming`. For a list of service IDs, see Table G–7.

Table 39–1 Site-Level Settings for Activity Stream

Setting Key	Description	Valid Site-Level Values
`accesscontrol.value.accessControlLevel`	Specifies who can view a user's Activity Stream.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`accesscontrol.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for who can view their Activity Stream. This setting is honored only when the application exposes the override control in a user preferences screen.	`+`, `-` `+`=TRUE `-`=FALSE

Table 39–1 (Cont.) Site-Level Settings for Activity Stream

Setting Key	Description	Valid Site-Level Values
`connectionsActivities.value`	Specifies whether the Home space activities of a user's connections are included in the user's Activity Stream. Users can override this setting on a task flow instance.	`NONE`, `ALL` `NONE`=No activities of the user's connections are shown. `ALL`=All activities of the user's connections are shown.
`groupSpaceActivities.value`	Specifies whether any space activities of a user's connections are included in the user's Activity Stream. Users can override this setting on a task flow instance.	`GSNONE`, `GSALL` `GSNONE`=No space activities are shown `GSALL`=Activities are shown for all of the spaces of which the user is a member and on which the user has view access.
`servicePublishedSettings[`*`serviceId`*`]`	Specifies whether to show activities published by the WebCenter service *`serviceId`* in a user's Activity Stream. A *`serviceId`* is the WebCenter service ID of any service that publishes activities. For a list of service IDs, see Table G–7. For example: `servicePublishedSettings[oracle.webcenter.community]` Users can override this setting on a task flow instance.	`+`, `-` `+`=TRUE `-`=FALSE

39.4.2.2 Connections Site-Level Settings

Table 39–2 lists and describes the site-level settings for the People Connections service Connections feature.

The service ID for Connections is `oracle.webcenter.peopleconnections.connections`. For a list of service IDs, see Table G–7.

Table 39–2 Site-Level Settings for Connections

Setting Key	Description	Valid Site-Level Values
`autoAcceptInvitations`	Specifies that connection invitations are accepted automatically by default. Individual users can override this application-level setting in their own view, provided the application exposes the override control in a user preferences screen.	`+`, `-` +=TRUE -=FALSE
`connectionsVisibility.accessControlSettings.accessControlLevel`	Specifies who can view a user's Connections.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`connectionsVisibility.personalizable`	Specifies whether individual users can override in their own application views the application-level setting for who can view their Connections. This setting is honored only when the application exposes the override control in a user preferences screen.	`+`, `-` +=TRUE -=FALSE

39.4.2.3 Feedback Site-Level Settings

Table 39–3 lists and describes the site-level settings for the People Connections service Feedback feature.

The service ID for Feedback is `oracle.webcenter.peopleconnections.kudos`. For a list of service IDs, see Table G–7.

Table 39–3 Site-Level Settings for Feedback

Setting Key	Description	Valid Site-Level Values
`visibility.accessControlSettings.accessControlLevel`	Specifies who can view a user's received Feedback.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`visibility.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for who can view their received Feedback. This setting is honored only when the application exposes the override control in a user preferences screen.	`+`, `-` +=TRUE -=FALSE
`addControl.accessControlSettings.accessControlLevel`	Specifies who can give Feedback to a user.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`addControl.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for who can give them Feedback. This setting is honored only when the application exposes the override control in a user preferences screen.	`+`, `-` +=TRUE -=FALSE

Table 39–3 (Cont.) Site-Level Settings for Feedback

Setting Key	Description	Valid Site-Level Values
`miniViewRowCount.value`	Specifies the number of Feedback entries to show in a Feedback - Quick View task flow.	Positive integers
`miniViewRowCount.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for the number of entries to show in a Feedback - Quick View task flow. This setting is honored only when the application exposes the override control in a user preferences screen.	+, – +=TRUE –=FALSE
`postedUserActions.delete`	Specifies whether users are allowed to delete the Feedback they leave for other users.	+, – +=TRUE –=FALSE

39.4.2.4 Message Board Site-Level Settings

Table 39–4 lists and describes the site-level settings for the People Connections service Message Board feature.

The service ID for Message Board is `oracle.webcenter.peopleconnections.wall`. For a list of service IDs, see Table G–7.

Table 39–4 Site-Level Settings For Message Board

Setting Key	Description	Valid Site-Level Values
`visibility.accessControlSettings.accessControlLevel`	Specifies who can view a user's Message Board.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`visibility.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for who can view their Message Board. This setting is honored only when the application exposes the override control in a user preferences screen.	+, – +=TRUE –=FALSE
`addControl.accessControlSettings.accessControlLevel`	Specifies who can post to a user's Message Board.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`addControl.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for who can post to their Message Board. This setting is honored only when the application exposes the override control in a user preferences screen.	+, – +=TRUE –=FALSE
`miniViewRowCount.value`	Specifies the number of messages to show in a Message Board - Quick View task flow.	Positive integer

Table 39–4 (Cont.) Site-Level Settings For Message Board

Setting Key	Description	Valid Site-Level Values
`miniViewRowCount.endUserConfigurable`	Specifies whether individual users can override in their own application views the application-level setting for the number of messages to show in a Message Board - Quick View task flow. This setting is honored only when the application exposes the override control in a user preferences screen.	`+`, `-` +=TRUE -=FALSE
`postedUserActions.edit`	Specifies whether users are allowed to edit the messages they post on other users' Message Boards.	`+`, `-` +=TRUE -=FALSE
`postedUserActions.delete`	Specifies whether users are allowed to delete messages they post on other users' Message Boards.	`+`, `-` +=TRUE -=FALSE

39.4.2.5 Profile Site-Level Settings

Table 39–5 lists and describes the site-level settings for the People Connections service Profile feature.

The service ID for Profile is `oracle.webcenter.peopleconnections.profile`. For a list of service IDs, see Table G–7.

Table 39–5 Site-Level Settings for Profile

Setting Key	Description	Valid Site-Level Values
`general-profile-settings.logActivity`	Specifies whether profile updates, such as uploading a photo, updating personal status, changing profile attributes, should result in activities getting published in Activity Stream.	`true`, `false`
`general-profile-settings.personalWebVisibility`	Specifies who can view a user's Profile Gallery.	`SELF`, `CONNECTIONS`, `USERS` (that is, authenticated users), `PUBLIC`
`general-profile-settings.pwVisibilityPersonalizable`	Specifies whether individual users can override in their own views the application-level setting for who can view their Profile Gallery. This setting is honored only when the application exposes the override control in a user preferences screen.	`true`, `false`
`general-profile-settings.view.name.`*`profileSectionName`*`.access-control-level`	Specifies who can view the Profile section identified by *`profileSectionName`*. For example: `general-profile-settings.view.name.personalInfo.access-control-level` For more information, see Table 39–6, " Profile Section Names".	`SELF`, `USERS` (that is, authenticated users), `PUBLIC` **Note:** `CONNECTIONS` is not a valid value for this setting.

Table 39–5 (Cont.) Site-Level Settings for Profile

Setting Key	Description	Valid Site-Level Values
`general-profile-settings.view.name.profileSectionName.allow-personalize-visibility`	Specifies whether individual users can override in their own application views the application-level setting for who can view the Profile section identified by *`profileSectionName`* on their profile. This setting is honored only when the application exposes the override control in a user preferences screen. For more information, see Table 39–6, " Profile Section Names".	`true`, `false`
`general-profile-settings.view.name.profileSectionName.allow-user-edit`	Specifies whether users are allowed to edit the Profile section identified by *`profileSectionName`* on their own Profiles. For example: `general-profile-settings.view.name.personalInfo.allow-user-edit` For more information, see Table 39–6, " Profile Section Names".	`true`, `false`
`ootb-view-edit-settings.view.name.profileSectionName.profileFieldName.allow-edit`	Specifies whether users are allowed to update the field identified by *`profileFieldName`* when they edit the Profile section identified by *`profileSectionName`* on their own profile. For example: `ootb-view-edit-settings.view.name.personalInfo.homePhone.allow-edit` For more information, see Table 39–6, " Profile Section Names" and Table 39–7, " Profile Field Names".	`true`, `false`

Table 39–6 Profile Section Names

Section Name	Description
`basic`	Profile summary These details are discoverable in an application search.
`employee`	Employee detail
`businessContact`	Business contact information
`personalInfo`	Personal information

Table 39–7 Profile Field Names

Field Name	Description
`email`	Business email address
`displayName`	User display name
`Dept`	Office department

Table 39–7 (Cont.) Profile Field Names

Field Name	Description
title	Job title
phone	Business phone number
timeZone	Time zone
photo	User photo
persStatNote	Personal status message
description	About me
empType	Employee type
empNo	Employee number
prefLang	User's preferred language
organization	Employee's organization
expertise	Employee's expertise
fax	Fax number
mobile	Mobile/cell phone number
pager	Page number
street	Office address: street
city	Office address: city
state	Office address: state
poBox	Office address: P.O. box
poCode	Office address: ZIP/PIN/P.O. code
country	Office address: country
homeAdd	Home address
homePhone	Home phone number
dob	Date of birth
maidenName	Maiden name/surname before marriage
doh	Date of hire

39.4.3 Example: Configuring Connections to Accept Invitations Automatically

By default, when users send invitations to connect, recipients must explicitly accept them. Through site-level settings, the application administrator can configure the application to accept invitations automatically. This setting is identified by the key, `autoAcceptInvitations`, and the service ID, `oracle.webcenter.peopleconnections.connections`. To change the setting, the administrator performs the following steps:

1. Open the `settings-insert.sql` script and uncomment the following lines:

```
-- INSERT INTO WC_PPL_COMMON_SETTING (ID, APPLICATION_ID, SCOPE_ID, SERVICE_ID,
USER_ID, TASKFLOW_INST_ID, SETTING_KEY, SETTING_VALUE)
--   VALUES ('768e5d1f-a73d-41b4-819d-74be081e1de1', 'webcenter',
'defaultScope', 'oracle.webcenter.peopleconnections.connections', 'SYSTEM',
'SITE', 'autoAcceptInvitations', '-')
 -- ;
```

After modification, the lines appear as follows:

```
 INSERT INTO WC_PPL_COMMON_SETTING (ID, APPLICATION_ID, SCOPE_ID, SERVICE_ID,
USER_ID, TASKFLOW_INST_ID, SETTING_KEY, SETTING_VALUE)
   VALUES ('768e5d1f-a73d-41b4-819d-74be081e1de1', 'webcenter', 'defaultScope',
'oracle.webcenter.peopleconnections.connections', 'SYSTEM', 'SITE',
'autoAcceptInvitations', '-')
 ;
```

2. Save and run the script.
3. Open the `settings-update.sql` script and uncomment the following lines:

```
  -- UPDATE WC_PPL_COMMON_SETTING SET SETTING_VALUE='-'
  --   WHERE APPLICATION_ID='webcenter' AND SCOPE_ID='defaultScope' AND USER_
ID='SYSTEM' AND TASKFLOW_INST_ID='SITE'
  --   AND SERVICE_ID='oracle.webcenter.peopleconnections.connections' AND
SETTING_KEY='autoAcceptInvitations'
   -- ;
```

After the modification, the lines appear as follows:

```
   UPDATE WC_PPL_COMMON_SETTING SET SETTING_VALUE='-'
     WHERE APPLICATION_ID='webcenter' AND SCOPE_ID='defaultScope' AND USER_
ID='SYSTEM' AND TASKFLOW_INST_ID='SITE'
     AND SERVICE_ID='oracle.webcenter.peopleconnections.connections' AND
SETTING_KEY='autoAcceptInvitations'
    ;
```

4. Change the value of `SETTING_VALUE` to a plus sign (+).

   After modification, the lines appear as follows:

```
   UPDATE WC_PPL_COMMON_SETTING SET SETTING_VALUE='+'
     WHERE APPLICATION_ID='webcenter' AND SCOPE_ID='defaultScope' AND USER_
ID='SYSTEM' AND TASKFLOW_INST_ID='SITE'
     AND SERVICE_ID='oracle.webcenter.peopleconnections.connections' AND
SETTING_KEY='autoAcceptInvitations'
    ;
```

5. Save and run the script.
6. If your WebCenter Portal: Framework application is running, restart the application.

   After you restart, the setting takes effect. In this example, when a user invites another user to connect, the connection is created automatically.

See Also: For lists and descriptions of People Connections site-level settings, see Section 39.4.2, "Supported Site-Level Settings for People Connections Features."

40

People Connections Task Flow Binding Parameters

Each People Connections task flow has a set of required and optional task flow binding parameters. These provide a means of capturing information that is useful to the task flow's successful function. For example, all People Connections task flows provide a binding parameter for capturing the ID of the current runtime user. This value (typically `#{securityContext.userName}`) enables the task flow to return People Connections data that is relevant to the current user.

In addition to providing required values for successful task flow rendering, you can use task flow binding parameters to customize the appearance and behavior of a task flow instance. For example, you can use parameter values to determine whether headers and footers are rendered, the number of rows and columns of information to show, whether to apply a filter to returned data, and the like.

You can provide task flow binding parameter values when you drag and drop a task flow onto an application page. Doing so opens the Task Flow Bindings dialog (for more information, see Section 39.2, "How to Add People Connections Task Flows to a Page"). You can also adjust task flow binding parameter values after you have placed a task flow on a page.

This chapter describes how to revise task flow parameter values at design time and provides a table that lists and describes the binding parameters associated with People Connections task flows. It includes the following sections:

- Section 40.1, "How to Revise People Connections Task Flow Binding Parameters"
- Section 40.2, "People Connections Task Flow Binding Parameters"
- Section 40.3, "What You Should Know About the Activity Stream Advanced Query Option"

40.1 How to Revise People Connections Task Flow Binding Parameters

After you have added a task flow to a page, you may want to customize the instance by revising its binding parameter values. This section describes how to access the Edit Task Flow Binding dialog and change binding parameter values.

To access the Edit Task Flow Binding dialog:

1. Open the application page and set the view to bindings by clicking the **Bindings** tab at the bottom of the page.
2. Under **Executables**, double-click the task flow for which to revise task flow binding parameters (Figure 40–1) to open the Edit Task Flow Binding dialog (Figure 40–2).

Tip: Task flow names under **Executables** differ from names in design view. For example, Message Board task flows are instead referred to as *wall*, and Feedback task flows are instead referred to as *kudos*.

Figure 40–1 Bindings View of an Application Page

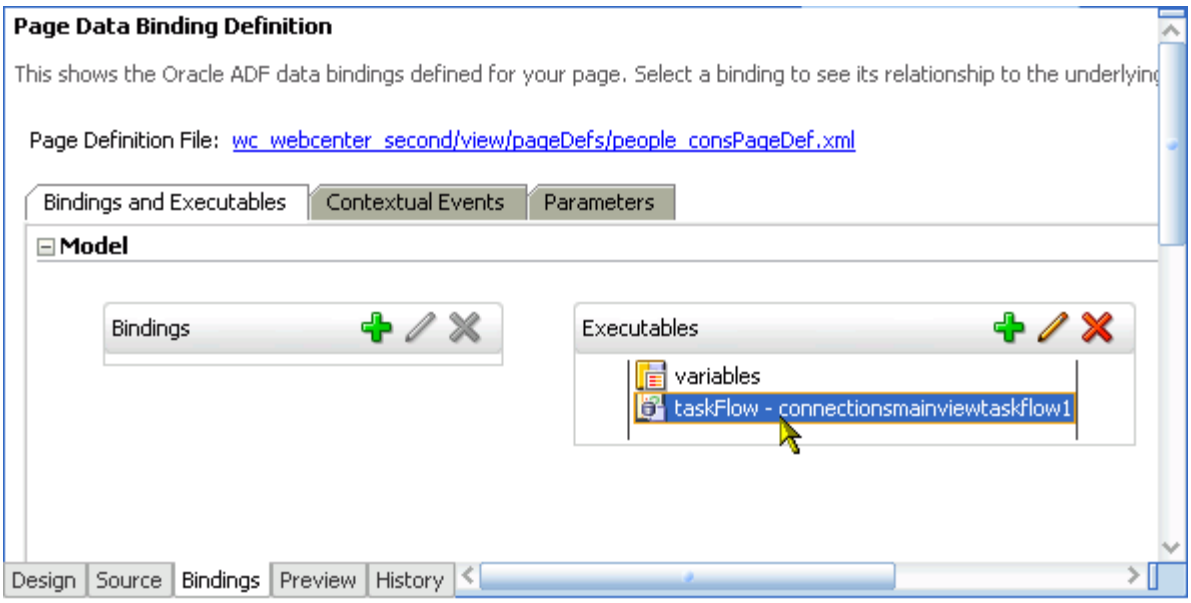
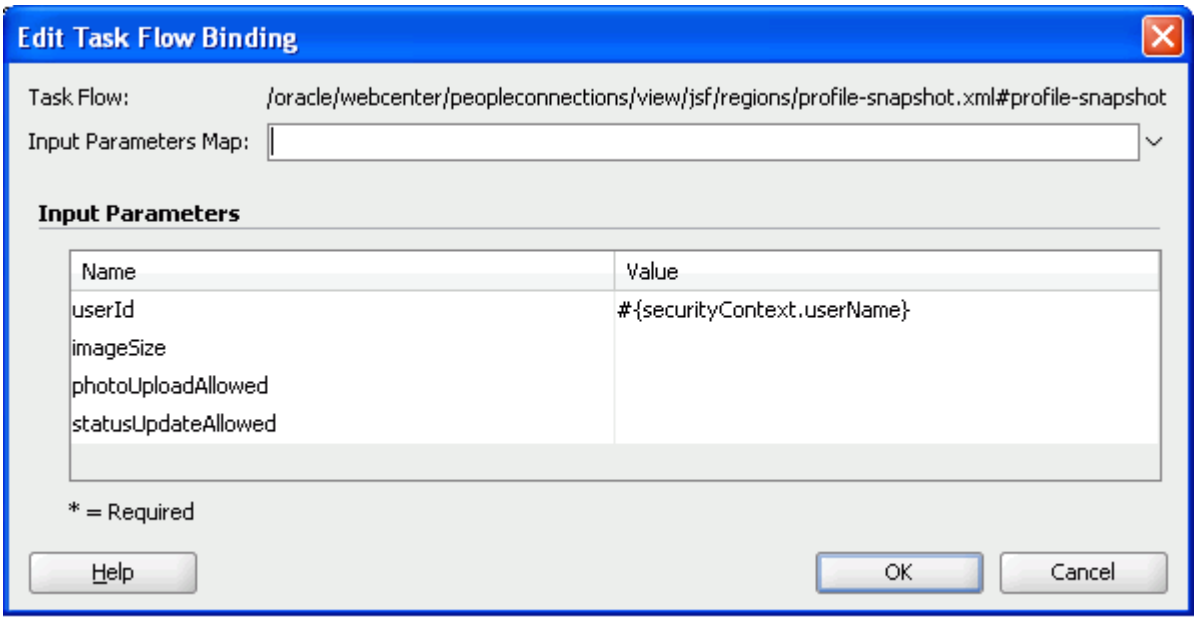


Figure 40–2 Edit Task Flow Binding Dialog



3. Revise binding parameter values (for more information, see Section 40.2, "People Connections Task Flow Binding Parameters").
4. Click **OK** to save your changes and exit the dialog.
5. Save and run your page to see the results.

40.2 People Connections Task Flow Binding Parameters

Table 40–1 lists and describes task flow binding parameters applicable to the People Connections service.

Table 40–1 People Connections Service Task Flow Binding Parameters

Parameter	Description
`__followEnforced`	A means of enforcing the follow logic when querying Activity Stream ■ Enter `#{true}` to establish that the user must follow the business object to access streamed activities, even when the object's activities are queried in an advance query clause. The default is true. ■ Enter `#{false}` to omit this requirement. The activities streamed for Fusion business objects are returned as far as required by the advanced query, no matter whether the user follows the objects. This parameter (for Fusion application business objects) is used in conjunction with the `advancedQuery` parameter. This parameter is associated with the Activity Stream task flow.
`__objectExtHandlerEnabled`	A means of using the object extension handler to process Fusion application business objects ■ Enter `#{true}` to use the object extension handler to process business objects. ■ Enter `#{false}` to omit using the object extension handler. False is the default. When you enter `#{true}`, then, before querying activities for the business objects, the Activity Stream uses the object extension handler to process the business objects. The handler checks whether the user has permission to access the business object. If the user does not have permission, the object is removed from the list, and, none activities are retrieved from this object. This parameter is associated with the following task flows: ■ Activity Stream ■ Activity Stream - Quick View
`__objects`	A comma separated list of objects for which to show activities Enter objects in the format `object-id;object-type;service-id`. Typically used only for followed objects. This parameter is associated with the Activity Stream task flow.
`__objectSecured`	A means of enforcing a security check on a Fusion application business object ■ Enter `#{true}` to enforce the security check. ■ Enter `#{false}` to omit the security check. False is the default. The security check is performed by the resource authorizer that is implemented by the Fusion application. If a user has no permission on a business object, the activities related to that business object are filtered out. This parameter is associated with the Activity Stream task flow. This parameter is associated with the following task flows: ■ Activity Stream ■ Activity Stream - Quick View
`advancedQuery`	A field for specifying a custom query to filter streamed items For more information, see Section 40.3, "What You Should Know About the Activity Stream Advanced Query Option." This parameter is associated with the Activity Stream task flow.

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`connectionListName`	The name of a connections list This parameter is associated with the task flow Connections - Card.
`currentView`	The view to display by default Valid values include: ▪ `connections`—(the default value) a list of connections ▪ `receivedInvitations`—a list of connections invitations you received ▪ `sentInvitations`—a list of connections invitations you sent ▪ `people`—a search field for finding people with whom to connect When users access the task flow instance, the view specified here is the first one they see. All selections, except `people`, provide controls for navigating to the application default view (`connections`). Selecting `people` provides search and select controls for inviting other users to connect. This parameter is associated with the Connections task flow.
`display_maxConnections`	The maximum number of connections to show in the task flow For example, enter `5` to specify that a maximum of five connections can appear in the task flow. A **More** link appears at the bottom of the task flow when there are more connections than the specified number of connections. Users click More to open the main view of the task flow where all connections are show. The value entered for this parameter is honored only when values for `display_numberOfRows` and `display_numberOfColumns` have not both been specified. If values for both of these parameters have been specified, then **Previous** and **Next** links appear when there are more connections than can fit into the specified numbers of rows and columns. This parameter is associated with the following task flows: ▪ Connections - Card ▪ Connections - Quick View
`display_numberOfColumns`	The number of columns to show in the task flow For example, in a Connections - Card task flow that shows six connections, a value of `2` means those connections are shown in two columns with three rows (see also `display_numberOfRows` and `display_maxConnections`). This parameter is associated with the following task flows: ▪ Connections - Card ▪ Connections - Quick View
`display_numberOfRows`	The number of rows to show in the task flow For example, in a Connections - Card task flow that shows six connections and a value of `2` for `display_numberOfColumns`, a value of `2` for this parameter means connections are shown in two columns with two rows. That is, four connections are shown. A **More** link appears at the bottom of the task flow when there are more connections than can be fit in the specified number of columns and rows. Users click More to open the main view of the task flow where all connections are shown. See also `display_maxConnections`. This parameter is associated with the following task flows: ▪ Connections - Card ▪ Connections - Quick View

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`display_profileFormat`	The default layout style for the task flow Enter one of the following: ■ `vcard`—displays a connection's photo and status message. ■ `iconic`—displays a connection's photo and user name. ■ `list`—displays connections in a list, with each list row showing the profile photo, user name (linked to a summary profile view), mail address, job title, status message, and actions icons and links. ■ `tiled`—renders the connection's personal Profile photo and shows the user name and job title beside the photo. In all layouts, the name links to a summary view of the user's Profile. This parameter is associated with the task flow Connections - Card.
`display_removalAllowed`	A means of showing or hiding the **Remove** action on the task flow Enter `true` to show the Remove action. Enter `false` to hide the Remove action. The default value (`true`) is applied if the input is invalid. This parameter is associated with the task flow Connections - Card.
`display_sortedBy`	The connections sort order Enter `LAST_ACTIVITY_TIME` to sort connections in descending date/time order. Leave this field blank to sort connections alphabetically by name. This parameter is associated with the task flow Connections - Card.
`displayCount`	The number of items to show in the task flow For example, enter `5` to specify that a maximum of five items can appear in the task flow. In quick views, a **More** link appears at the bottom of the task flow when there are more items than the specified number of items. Users click **More** to open the main view of the task flow where all items are accessible. In main views, **Previous** and **Next** links are shown. Users click these to page through entries. This parameter appears in the component properties for the following task flows: ■ Feedback ■ Feedback - Quick View ■ Message Board ■ Message Board - Quick View
`displayMessageSize`	The number of characters to show for each feedback message Messages exceeding the specified value are truncated, and an ellipse (…) is appended to the end. This parameter is associated with the task flow Feedback - Quick View.
`editMode`	A means of enabling an edit mode on a Profile task flow instance Enter `#{true}` to enable an edit mode. Enter `#{false}` to disable it. This parameter is associated with the Profile task flow.

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`enableContextInfo`	A means of showing or omitting detailed information about the object in the current context (that is, in a popup or other contextual instrument) ■ Enter `#{true}` to enable Activity Stream to display the `af:contextInfo` component (a small, red dot). Users can click this dot to view object detail information. ■ Enter `#{false}` to omit object detail information. When this parameter is true and the `contextInfoTaskflowId` in the service definition is defined, the activity from the service displays the `af:contextInfo`. Otherwise, no context information is shown. This parameter is associated with the Activity Stream task flow.
`filterPattern`	A filter against task flow content For example, enter `pat` to show only those connections named *pat* (including *patrick* or *sripathy*). This parameter is associated with the task flow Connections - Card.
`fromDate`	The start date of a date range within which to show feedback messages Enter dates in the format `YYYY-MM-DD`. See also `toDate`. This parameter is associated with the following task flows: ■ Feedback ■ Feedback - Quick View ■ Message Board
`hideActions`	A means of hiding the actions normally associated with a Feedback or Message Board entry, such as Edit, Private, Hide, and Delete ■ Enter `#{true}` to hide actions associated with a Feedback or Message Board entry. ■ Enter `#{false}` to show such actions. When no value is entered, `false` is the default. This parameter appears in the component properties for the following task flows: ■ Feedback – Quick View ■ Message Board – Quick View
`hideAttach`	Specifies whether the **Attach: File | Link** option is shown or hidden ■ Check for `#{true}` to hide the **Attach: File | Link** option. ■ Clear for `#{false}` to show the **Attach: File | Link** option. This parameter is associated with the Publisher task flow.
`hideComments`	A means of showing or hiding the Comments feature on streamed activities Enter `true` to hide the Comments feature. Enter `false` to show it. This parameter is associated with the Activity Stream task flow
`hideConfigure`	A means of hiding the personalization option on the task flow instance This parameter is associated with the following task flows: ■ Activity Stream ■ Activity Stream - Quick View

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`hideDocumentUploader`	A means of showing or hiding the Publisher task flow's document uploader Enter `#{true}` to hide the Publisher task flow's document uploader. Enter `#{false}` to show it. This parameter is associated with the Publisher task flow.
`hideFooter`	A means of showing or hiding the task flow footer The task flow footer contains the **More** link that appears at the bottom of a task flow when there are more items to show than can be accommodated in the current view. Hiding the footer hides the **More** link. Hiding the footer does not affect **Previous** and **Next** links that may also appear at the bottom of a task flow. Enter `true` to hide the footer. Enter `false` (default) to show the footer. This parameter is associated with the following task flows: ■ Activity Stream - Quick View ■ Connections - Card ■ Connections - Quick View ■ Feedback - Quick View ■ Message Board - Quick View
`hideGiven`	A means of showing or hiding feedback given to a user Enter true to allow the rendering of given feedback. Enter false to prohibit it. This parameter is associated with the following task flows: ■ Feedback ■ Feedback - Quick View
`hideHeader`	A means of showing or hiding the task flow header Enter `true` to hide the task flow header. Enter `false` (default) to show the task flow header. This parameter is associated with the following task flows: ■ Activity Stream - Quick View ■ Connections - Quick View ■ Feedback ■ Feedback - Quick View ■ Message Board ■ Message Board - Quick View
`hideInlinePreview`	A means of allowing or omitting an inline preview of files attached to streamed activities Enter `true` to omit a file preview. Enter `false` to show it. This parameter is associated with the Activity Stream task flow.
`hideLike`	A means of showing or hiding the Like link on a streamed activity Enter true to hide the Like link. Enter false to show it. This parameter is associated with the Activity Stream task flow

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`hidePublisher`	A means of showing or hiding the message entry field and the upload file and URL controls (the Publisher) ■ Enter `#{true}` to disable the display of the message entry field in a given task flow instance. ■ Enter `#{false}` to allow the display of the message entry field in a given task flow instance. This parameter is associated with the following task flows: ■ Message Board ■ Message Board - Quick View
`hideScopePicker`	Specifies whether the **Share something with** option is shown or hidden ■ Check for `#{true}` to hide the **Share something with** option. ■ Clear for `#{false}` to show the Share something with option. This parameter is associated with the Publisher task flow.
`hintsTextKey`	Specifies the resource bundle class and message key for hint text Use the format `key[,RBClass]. __EMPTY__` as the predefined key for no hint text. This is the default value. This parameter is associated with the Publisher task flow.
`imageSize`	The display size of the profile photo Enter one of the following values: ■ `ORIGINAL` ■ `LARGE` ■ `MEDIUM` ■ `SMALL` This parameter is associated with the Profile - Snapshot task flow.
`isUpdateStatus`	A means of dedicating the task flow instance to updating the current user's Profile status message Enter `#{true}` to devote the task flow instance to updating user status messages. Enter `#{false}` to omit publishing messages to the current user's Profile status message. This parameter is associated with the Publisher task flow.
`keepOpenAfterPublish`	Specifies whether the text box remains active after a user clicks the **Publish** button ■ Enter `#{true}` to keep the text box open after a message is published. ■ Enter `#{false}` to close the text box after a message is published. This parameter is associated with the Publisher task flow.
`numberOfRows_list`	The number of rows to show when the Connections task flow instance is set to list view This parameter is associated with the Connections task flow.
`objectId`	The ID of an object to associate with a published message This parameter is associated with the Publisher task flow.
`objectType`	The type of an object to associate with a published message This parameter is associated with the Publisher task flow.

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`orgBreadCrumbsShown`	A means of showing or omitting organization breadcrumbs on a user profile that renders a management chain that is linked to each member up the chain Enter `#{true}` to show organizational breadcrumbs. Enter `#{false}` to omit them. This parameter is associated with the Profile - Snapshot task flow.
`pageSize`	The number of items to stream in a given task flow instance This parameter is associated with the following task flows: ■ Activity Stream ■ Activity Stream - Quick View
`pagination`	The form of pagination to use on a multipage stream ■ Enter `#{true}` to show Previous and Next links. ■ Enter `#{false}` hide to omit Previous and Next links. Instead, a more link is rendered, enabling users to navigate to a fuller view of the task flow where all streamed activities are shown. This parameter is associated with the Activity Stream task flow.
`photoUploadAllowed`	A means of allowing a photo upload from the profile screen Enter `true` (default) to allow users to upload a photo to their profile. Enter `false` to prevent users from uploading a photo. This parameter is associated with the task flow Profile - Snapshot.
`profileEditLinkShown`	A means of showing or omitting an Edit link on a Profile - Snapshot task flow Enter `#{true}` to show the Edit link. Enter `#{false}` to hide it. This parameter is associated with the Profile - Snapshot task flow.
`profileLaunchedOnSelection`	A means of launching a user Profile when the user name is clicked Enter `true` to launch a user Profile when the user name is clicked. Enter `false` to prevent launching a user Profile This parameter is associated with the task flow Connections - Card
`profileOnly`	A means of streaming activities only from user profiles ■ Enter `true` to stream only those activities associated with user profiles. ■ Enter `false` to stream other types of activities along with those associated with user profiles. This parameter is associated with the Activity Stream task flow.
`profileUserId`	ID of the user from whom to stream activity This parameter is associated with the Activity Steam - Quick View task flow.

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`resourceId`	The current user ID Enter `#{securityContext.userName}` to return the current user. This parameter is associated with the following task flows: ■ Activity Stream ■ Connections ■ Feedback ■ Message Board ■ Profile ■ Profile Gallery
`scopeId`	The ID of the scope to which to publish This property value assists in generating a link for use in navigating to the published object. It is not necessary to provide a value, unless you plan to do so using an EL expression. For information about EL expressions, seeAppendix G, "Expression Language Expressions." This parameter is associated with the Publisher task flow.
`sectionWiseEditEnabled`	A means of enabling or disabling section-by-section edit capability on the task flow ■ Enter `#{true}` to enable section-by-section editing. An **Edit** control is shown on each editable Profile section. ■ Enter `#{false}` to disable section-by-section editing. This parameter is associated with the Profile task flow.
`serviceCategories`	A field for entering a comma-separated list of names of services from which to stream activities Use this parameter to limit the display of streamed activities to only those associated with the specified service or services. Enter one or more service IDs, for example: `oracle.webcenter.collab.announcement, oracle.webcenter.collab.forum` For a list of valid service ID, see Table G–7, " Service IDs". Note that all listed service IDs cannot be used because all services do not stream items to the Activity Stream. For example, the RSS service does not stream any activities. This parameter is associated with the Activity Stream task flow.
`serviceId`	The service ID of the service to which the object associated with a published message belongs For a list of service IDs, see Table G–7, " Service IDs". This parameter is associated with the Publisher task flow.
`spaceName`	The space to which to publish messages If your application does not support spaces, do not provide a value for this parameter. This parameter is associated with the following task flows: ■ Message Board ■ Message Board - Quick View ■ Publisher

Table 40–1 (Cont.) People Connections Service Task Flow Binding Parameters

Parameter	Description
`spaces`	A comma-separated list of names or GUIDs of spaces from which to stream activities Use this parameter to limit the display of streamed activities to only those associated with the specified space or spaces. If your application does not support spaces, do not provide a value for this parameter. This parameter is associated with the Activity Stream task flow.
`statusUpdateAllowed`	A boolean value representing whether a control is available on the task flow for updating a profile status message Enter either `true` or `false`. This parameter appears in the component properties for the Profile - Snapshot task flow.
`toDate`	The end date of a date range within which to show feedback messages Enter dates in the format `YYYY-MM-DD`. See also `fromDate`. This parameter is associated with the following task flows: ■ Feedback ■ Feedback - Quick View ■ Message Board
`uploadDocumentOnly`	A means of limiting the Publisher task flow to its document upload feature Enter #{true} to limit the Publisher task flow to be only a document uploader. Enter `#{false}` to expose all Publisher features. This parameter is associated with the Publisher task flow.
`userId`	The current user ID Enter `#{securityContext.userName}` to return the current user. This parameter is associated with the following task flows: ■ Connections - Card ■ Connections - Quick View ■ Feedback - Quick View ■ Message Board - Quick View ■ Profile - Snapshot
`userName`	The name of the user to whom to publish messages This value is supplied by default. We recommend that you do not change the default value, `#{o_w_w_i_v_b_resourceViewerBean.username}`. This parameter is associated with the Publisher task flow.
`viaUser`	The user name of the person who provided the object the current user is sharing For example, John is sharing a document with everyone that Jane originally shared with him. In this case, Jane is the via user. This parameter is associated with the Publisher task flow.

40.3 What You Should Know About the Activity Stream Advanced Query Option

Use `Advanced Query` to create filters against streamed activities in an Activity Stream task flow. Create filters for user names, service IDs, and object details, such as a document's display name. You can use SQL syntax for parameter values. Additionally you can place EL expressions within the SQL.

You can construct queries against specific database objects, which are represented by aliases that are prefixed to the inquiry. Table 40–2 lists and describes the types of database objects against which you can construct a query and provides their alias prefixes.

> **See Also:** In many cases, you can use EL expressions to obtain the value you require for the supported fields and columns listed in Table 40–2. For more information, see Appendix G, "Expression Language Expressions."

Table 40–2 Supported Database Objects for Constructing a SQL WHERE Clause

Database Object	Alias Prefix	Supported Fields/Columns
ACTIVITY	AE	■ `SCOPE_ID`—The GUID of the scope (you can use an EL to return this value, for example, `#{serviceCtx.scope.GUID}`). ■ `SERVICE_ID`—The service ID of the service to track. See Table G–7 for a list of service IDs. ■ `ACTIVITY_TIME`—The time the activity occurs. Use the datetime format that is supported in the SQL or target database. You can also use Oracle database SQL constructs, such as `to_date()`. ■ `ACTIVITY_TYPE`—The type of activity to track For a list of valid activity type names, see Table 40–3.
ACTIVITY (ACTOR)	AD	`ACTOR_NAME`—The user name of the person performing the activity.
ACTIVITY (OBJECT)	OD	■ `SERVICE_ID`—The service ID of the service from which the tracked object issues. See Table G–7 for a list of service IDs. ■ `OBJECT_ID`—The GUID of the object. ■ `DISPLAY_NAME`—The object display name. ■ `OBJECT_TYPE`—The object type. Object type names for use with Advanced Query include: ■ `event` ■ `announcement` ■ `forum` ■ `topic` ■ `bookmark` ■ `list` ■ `page` ■ `blog` ■ `document` ■ `wiki`

Table 40–3 Activity Type Names for Advanced Query

Service	Activity Type Name
Events	▪ createEvent ▪ updateEvent
Announcements	▪ createAnnouncement ▪ updateAnnouncement
Discussions	▪ createForum ▪ createTopic ▪ replyTopic
Tags	▪ updateBookmark
Lists	▪ createList ▪ editList
Page	▪ createPage ▪ editPage
Documents	▪ create-blog ▪ update-blog ▪ create-document ▪ create-wiki ▪ update-document ▪ update-wiki
People Connections (Profile)	▪ updateStatus ▪ updateProfile ▪ updatePhoto
People Connections (Message Board)	▪ postScope ▪ postself ▪ post ▪ sharescope ▪ shareself ▪ share ▪ shareobjectscope ▪ shareobjectself ▪ shareobject ▪ updatescope ▪ updateself ▪ update
People Connections (Connections)	▪ connect ▪ inviteForConnection
People Connections (Feedback)	▪ post

The SQL string that is passed as the advanced query parameter complies with SQL standards. That is, it supports SQL constructs, such as `AND`, `OR`, `IN`, and the like. Note, however, that it does not support `INSERT`, `UPDATE`, `DELETE`, `SELECT`, `JOIN` constructs. The syntax of the advanced query must contain only the `WHERE` clause

portion of a SQL query. Because `SELECT` is not supported, the `WHERE` clause cannot have nested queries or subqueries.

The `Advanced Query` parameter also supports EL expressions, which can be embedded in the `WHERE` clause or used to generate the whole `WHERE` clause.

All the literals in the query must be escaped by prepending a backward slash (\); otherwise, such characters generate syntax errors (see Table 40–4 for examples).

Note that the advanced query `WHERE` clause is always `AND`'ed to the internal query that is generated by Activity Stream based on the current user, space membership, connection list, and the like. This is to prevent a user from viewing activities to which he or she does not have access.

Table 40–4 lists examples of advanced queries.

Table 40–4 Examples of Advanced Queries for Use with Activity Stream

Use Case	Query Syntax
Stream only document creation activities.	`AE.SERVICE_ID = \'oracle.webcenter.doclib\'`
Stream activities only from an object or current space.	`OD.OBJECT_ID = \'objectA\' OR AE.SCOPE_ID = \'#{serviceCtx.scope.GUID}\'`
Stream activities only about wikis created by the current user.	`OD.OBJECT_TYPE = \'Wiki\' AND AD.ACTOR_NAME = \'#{securityContext.userName}\'`
Stream activities for documents and discussions, but only create activities or all activities for the current user.	`(AE.SERVICE_ID IN (\'oracle.webcenter.doclib\', \'oracle.webcenter.collab.forum\') AND AE.ACTIVITY_TYPE IN (\'createDocument\', \'createTopic\')) OR AD.ACTOR_NAME = \'#{securityContext.userName}\'`

41

Using People Connections Data Controls and Java APIs

This chapter provides an overview of People Connections service data controls and Java APIs. It further provides details about each data control and how to add it to your project and a pointer to information about how to use data controls in a WebCenter Portal: Framework application.

This chapter includes the following sections:

- Section 41.1, "Using People Connections Data Controls"
- Section 41.2, "People Connections Service Java APIs"

41.1 Using People Connections Data Controls

The People Connections service provides data controls that enable you to create your own visualization of the People Connections functionality. This section provides an overview of the following data controls and lists and describes their supported methods, attributes, and classes:

- People Connections Management Data Control
- Kudos Service Data Control
- Profile Data Control

This section also includes the following subsections:

- Section 41.1.1, "Adding a Data Control to Your Project"
- Section 41.1.2, "Working with People Connections Management Data Control"
- Section 41.1.3, "Working with Profile Data Control"

41.1.1 Adding a Data Control to Your Project

You add a data control to your project by right-clicking it in the Resource Palette and selecting **Add to Project** from the resulting context menu. Once added, you can browse the data control's methods and attributes by expanding it in the Data Controls panel in the Application Navigator.

You add a data control to an application page by dragging it onto the page from the Data Controls panel. Once placed, a context menu opens with options for selecting the type of component to which to bind the data, such as a button or a text box.

To implement a People Connections Data Control:

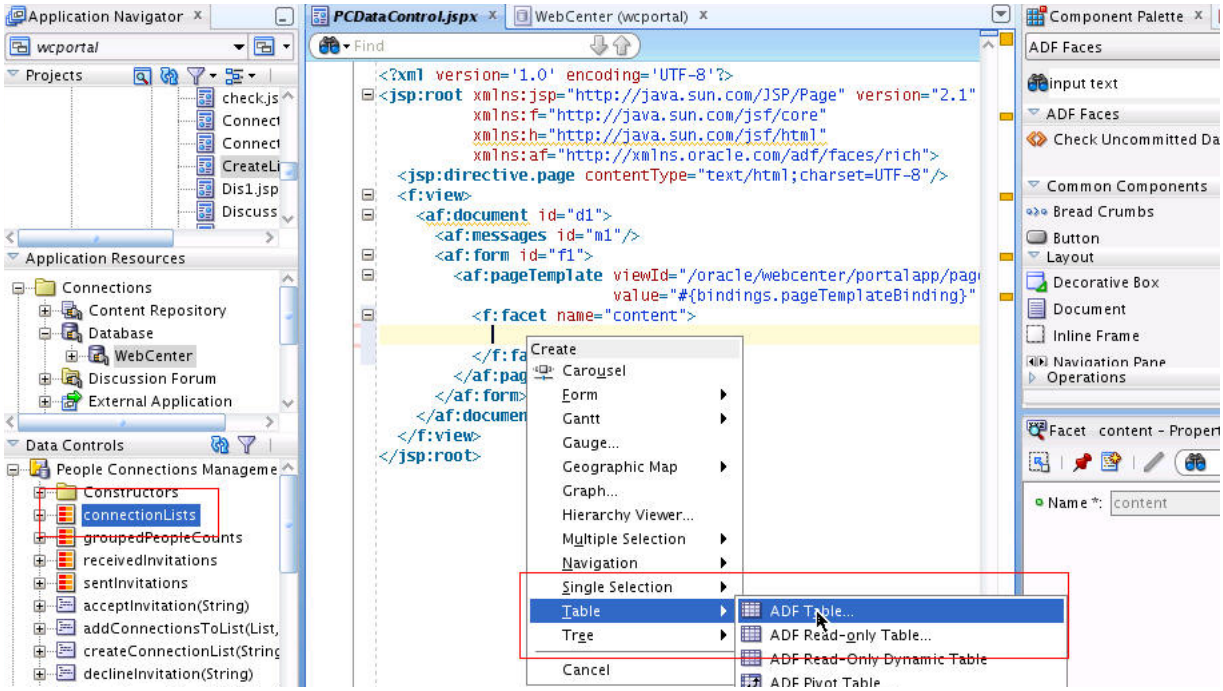
1. Create a WebCenter Portal: Framework application in JDeveloper..

2. In your WebCenter portal application, navigate to **Resource Catalog** > **Services Catalog** > **Data Controls**.
3. Select a People Connection Data Control and right-click the control.
4. Click **Add to Project** in the context menu.

   The Data Controls appear in the Application Navigator Projects Data Controls folder.
5. Expand the Data Controls folder.

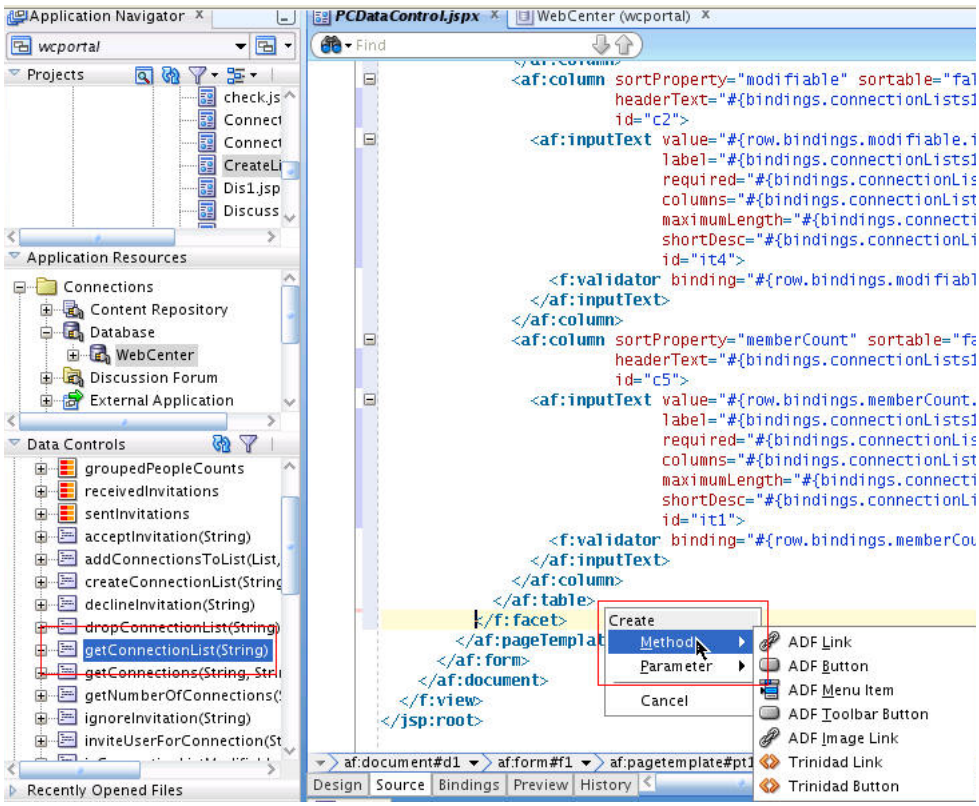
   On expanding, the Data Control objects and their corresponding Java methods appear.
6. Drag and drop the data control object on to the `.jspx` page and select the **ADF Table** option in the drop-down **Table** menu list, shown in Figure 41–1.

Figure 41–1 The Data Control Object Dragged and Dropped on to a .jspx Page With ADF Table Selected



7. To use the available Java methods, drag the methods to the .jspx page and select the **Method** item in the **Create** menu list, as shown in Figure 41–2, and bind it to the data in your project to achieve the desired functionality.

Figure 41–2 Dragging and Dropping Java Methods to the .jspx Page and Selecting Method in the Create Menu List



See Also: For more information about using data controls in a WebCenter Portal: Framework application, see Section 7.1.3, "Using WebCenter Portal Data Controls."

41.1.2 Working with People Connections Management Data Control

The People Connections Management Data Control provides methods for adding and managing connections and connections lists. The primary methods exposed include:

- connectionLists
- groupedPeopleCounts
- receivedInvitations
- sentInvitations
- acceptInvitation(String invitationId)
- addConnectionsToList(List userids, String listName)
- createConnectionList(String listName)
- declineInvitation(String invitationId)
- dropConnectionList(String listName)
- getConnectionList(String connectionListName)
- getConnections(String userid, String connectionListName, String filterPattern, String sortBy, int startIndex, int fetchSize)

- getNumberOfConnections(String userid, String connectionListName, String filterPattern)
- ignoreInvitation(String invitationId)
- inviteUserForConnection(String userid, String invitationMessage, Collection connectionListNames)
- isConnectionListModifiable(String listName)
- isConnectionPartOfUnmodifiableList(String connecteeGuid)
- removeConnection(String userid, boolean fromListOnly, String listName)
- searchUsers(String filterPattern, int startIndex, int fetchSize)
- updateConnectionListsMembership(String member, Collection addListIds, Collection removeListIds)

The subsections in this section list and describe the methods associated with the People Connections Management Data Control.

41.1.2.1 connectionLists

A collection of details about connections lists, including the list ID, name, localized name, number of members, whether or not the list is modifiable, and member user IDs. Table 41–1 lists and describes the attributes associated with this method.

Table 41–1 Attributes of the Method connectionLists

Attribute	Description
`id`	The connections lists IDs
`listName`	The names of connections lists
`localizedName`	The localized names of connections lists
`memberCount`	The number of members on each named connections list
`modifiable`	Information about whether a named connections list can be modified
`memberUserIds`	The user IDs of members on each named connections list

41.1.2.2 groupedPeopleCounts

The number of connections contained in each connections group.

41.1.2.3 receivedInvitations

A collection of details about a received invitation to connect, including when the invitation was sent and received, the invitation's GUID, the invitation message, and the user IDs of the user who sent the invitation and who received it. Table 41–2 lists and describes the attributes associated with this method.

Table 41–2 Attributes of the Method receivedInvitations

Attribute	Description
`dateFormattedAsReceived`	The date the invitation was received according to the recipient's time stamp
`dateFormattedAsSent`	The date the inviation was sent according to the sender's time stamp.
`id`	The GUID of the received invitation
`invitationMessage`	The invitation message

Table 41–2 (Cont.) Attributes of the Method receivedInvitations

Attribute	Description
inviteeUserid	The user ID of the person who received the invitation
invitorUserid	The user ID of the person who sent the invitation
sentDate	The date the invitation was sent according to the system time stamp

41.1.2.4 sentInvitations

A collection of details about a sent invitation to connect, including when the invitation was sent and received, the invitation's GUID, the invitation message, and the user IDs of the user who sent the invitation and who received it. lists and describes the attributes associated with this method. Table 41–3 lists and describes the attributes associated with this method.

Table 41–3 Attributes of the Method sentInvitations

Attribute	Description
dateFormattedAsReceived	The date the invitation was received according to the recipient's time stamp
dateFormattedAsSent	The date the inviation was sent according to the sender's time stamp
id	The unique GUID of the received invitation
invitationMessage	The invitation message
inviteeUserid	The user ID of the person who received the invitation
invitorUserid	The user ID of the person who sent the invitation
sentDate	The date the invitation was sent according to the system time stamp

41.1.2.5 acceptInvitation(String invitationId)

A method for accepting invitations to connect. Table 41–4 lists and describes the parameters associated with this method.

Table 41–4 Parameters of the Method acceptInvitations

Parameter	Description
invitationId	The GUID of the invitation to connect to be accepted

41.1.2.6 addConnectionsToList(List userids, String listName)

A method for adding identified users to a named connections list. Table 41–5 lists and describes the parameters associated with this method.

Table 41–5 Parameters of the Method addConnectionsToList

Parameter	Description
userids	The user names of the users to add to the named connections list
listName	The name of the connections list to which to add identified users

41.1.2.7 createConnectionList(String listName)

A method for creating a connections list. Table 41–6 lists and describes the parameters associated with this method.

Table 41–6 Parameters ofthe Method createConnectionList

Parameter	Description
`listName`	The name to give the new connections list

41.1.2.8 declineInvitation(String invitationId)

A method for declining an invitation to connect. Table 41–7 lists and describes the parameters associated with this method.

Table 41–7 Parameters of the Method declineInvitation

Parameter	Description
`invitationId`	The unique GUID of the declined invitation

41.1.2.9 dropConnectionList(String listName)

Method for deleting a connections list. Table 41–8 lists and describes the parameters associated with this method.

Table 41–8 Parameters of the Method dropConnectionList

Parameter	Description
`listName`	The name of the connections list to delete

41.1.2.10 getConnectionList(String connectionListName)

Returns an object that represents a Connections List. This object provides details about a connections list acquired for the given connection list name. Table 41–9 lists and describes the parameters associated with this method. Table 41–10 lists and describes the attributes associated with this method.

Table 41–9 Parameters of the Method getConnectionList

Parameter	Description
connectionListName	The name of the connections list for which to return the following attributes (see Table 41–10): ▪ `id` ▪ `listName` ▪ `localizedName` ▪ `memberCount` ▪ `modifiable` ▪ `memberUserIds`

Table 41–10 Attributes of the Method getConnectionList

Attribute	Description
`id`	The unique GUID of the connections list to return
`listName`	The name of the connections list to return
`localizedName`	The localized name of the connections list to return
`memberCount`	The number of members on the connections list to return
`modifiable`	An indicator of whether the returned connections list can be modified
`memberUserIds`	The user IDs of the members of a returned connections list

41.1.2.11 getConnections(String userid, String connectionListName, String filterPattern, String sortBy, int startIndex, int fetchSize)

Returns the `Connection` object, which provides details about the connection, including the connection's user ID, the connections lists on which the connection is a member, and details from a connection's profile. Table 41–11 lists and describes the parameters associated with this method. Table 41–12 lists and describes the attributes associated with this method.

Table 41–11 Parameters of the Method getConnections

Parameter	Description
`userid`	The user ID for each returned connection
`connectionListName`	The name of the connection list from which to obtain connections
`filterPattern`	A filter to use against returned connections, such as "co," which returns `condoleezza` as well as `nicole`
`sortBy`	The order by which to sort connections Enter `LAST_ACTIVITY_TIME` to sort connections in descending date/time order. Leave blank to sort alphabetically by name.
`startIndex`	The point from which to start fetching results This is used for pagination. For example, the search can return 50 matching records, and only 10 are needed, starting from 1. Set `startIndex` to 1 and `fetchSize` to 10.
`fetchSize`	The number of results to return (see `startIndex`)

Table 41–12 Attributes of the Method getConnections

Attribute	Description
`connecteeUserId`	The user ID of a connection
`connectionListsMembershipList`	The names of connections lists of which the connection is a member
`connecteeUserProfile`	The connectee's Profile details

41.1.2.12 getNumberOfConnections(String userid, String connectionListName, String filterPattern)

Returns the number of users (`int`) connected to the identified user. Table 41–13 lists and describes the parameters associated with this method. Table 41–14 lists and describes the attributes associated with this method.

Table 41–13 Parameters of the Method getNumberOfConnections

Parameter	Description
`userid`	The user ID for whom to return the number of connections
`connectionListName`	The name of the connection list from which to obtain a count
`filterPattern`	A filter to use against returned connections, such as "co," which returns `condoleezza` as well as `nicole`

Table 41–14 Attributes of the Method getNumberOfConnections

Attribute	Description
`Int`	The number of connections to return

41.1.2.13 ignoreInvitation(String invitationId)

Specifies that the invitation with the specified ID should be ignored. Table 41–15 lists and describes the parameters associated with this method.

Table 41–15 Parameters of the Method ignoreInvitation

Parameter	Description
`invitationId`	The GUID of the connection invitation to ignore

41.1.2.14 inviteUserForConnection(String userid, String invitationMessage, Collection connectionListNames)

A means of inviting the user with the specified user ID to connect. Includes the invitation message, such as, "I would like you to be my connection." Also includes one or more connection list names to which to add the invitee once the invitation is accepted. Table 41–16 lists and describes the parameters associated with this method.

Table 41–16 Parameters of the Method inviteUserForConnection

Parameter	Description
`userid`	User ID of the invitee
`invitationMessage`	Connection message, such as, "Let's connect!"
`connectionListNames`	The names of the connecitons lists to add the invitee to once the invitation is accepted

41.1.2.15 isConnectionListModifiable(String listName)

Returns true or false (`boolean`) as to whether the named connections list can be modified. Table 41–17 lists and describes the parameters associated with this method. Table 41–18 lists and describes the attributes associated with this method.

Table 41–17 Parameters of the Method isConnectionListModifiable

Parameter	Description
`listName`	The name of the connections list for which to determine whether it can be modified

Table 41–18 Attributes of the MethodisConnecitonListModifiable

Attribute	Description
`boolean`	True or false, depending on whether the named connections list can be modified

41.1.2.16 isConnectionPartOfUnmodifiableList(String connecteeGuid)

Returns true or false (boolean) as to whether the identified user is a member of a connections list that cannot be modified. Table 41–19 lists and describes the parameters associated with this method.

Table 41–19 Parameters of the Method isConnectionPartOfUnmodifiableList

Parameter	Description
`connecteeGuid`	The GUID of the connection for whom to determine whether this connection is a member of a connections list that cannot be modified

41.1.2.17 removeConnection(String userid, boolean fromListOnly, String listName)

A method for removing the identified connection either as a connection or from a connections list. Table 41–20 lists and describes the parameters associated with this method.

Table 41–20 Parameters of the Method removeConnection

Parameter	Description
`userid`	The user name of the connection to remove from the specified list
`fromListsOnly`	A flag (true or false) indicating whether to remove the user as a connection or remove the user only from the specified connections list
`listName`	The name of the connections list from which to remove the specified user

41.1.2.18 searchUsers(String filterPattern, int startIndex, int fetchSize)

Returns the object `Users`, which is a collection of details about returned users, including the user's relational attributes, GUID, and profile details. Table 41–21 lists and describes the parameters associated with this method. Table 41–22 lists and describes the attributes associated with this method.

Table 41–21 Parameters of the Method searchUsers

Parameter	Description
`filterPattern`	A filter to use against returned search terms, such as "co," which returns `condoleezza` as well as `nicole`
`startIndex`	The point from which to start fetching results This is used for pagination. For example, the search can return 50 matching records, and only 10 are needed, starting from 1. Set `startIndex` to 1 and `fetchSize` to 10.
`fetchSize`	The number of results to return (see `startIndex`)

Table 41–22 Attributes of the Method searchUsers

Attribute	Description
`relationalAttributes`	—
`userGuid`	The user's unique GUID
`userProfile`	The user's Profile details

41.1.2.19 updateConnectionListsMembership(String member, Collection addListIds, Collection removeListIds)

Provides a means of updating an identified connection's list membership by adding the connection to identified connections lists and removing the connection from identified connections lists. Table 41–23 lists and describes the parameters associated with this method.

Table 41–23 Parameters of the updateConnectionListsMembership Method

Parameter	Description
`member`	The user name of the member whose connections list membership to manage

Table 41–23 (Cont.) Parameters of the updateConnectionListsMembership Method

Parameter	Description
`addListIds`	The names of the connections lists to which to add the identified user
`removeListIds`	The names of the connections lists from which to remove the identified user

41.1.3 Working with Profile Data Control

The Profile Data Control provides methods for returning and updating Profile details.

The subsections in this section list and describe the methods associated with the Profile Data Control.

41.1.3.1 Method: getProfile(String userId)

A method for returning the `WCUserProfile` object, which is a collection of user Profile details. Table 41–24 lists and describes the parameters associated with this method.

Table 41–24 Parameters for the getProfile Method

Parameter	Description
`userID`	The ID of the user for whom to return Profile details

> **Tip:** To see the attributes associated with the `WCUserProfile` object, expand the `WCUserProfile` node under the `getProfile` method on the Data Controls panel in the Application Navigator.

41.1.3.2 Method: getProfileForUpdate(String userId)

A method for updating the `WCUserProfile` object, which is a collection of user Profile details. Table 41–25 lists and describes the parameters associated with this method.

Table 41–25 Parameters for the getProfileForUpdate Method

Parameter	Description
`userID`	The ID of the user for whom to update Profile details

> **Tip:** To see the attributes associated with the `WCUserProfile` object, expand the `WCUserProfile` node under the `getProfileForUpdate` method on the Data Controls panel in the Application Navigator.

41.2 People Connections Service Java APIs

The People Connections service has associated Java APIs that you can use to work with service features. These include:

- `oracle.webcenter.peopleconnections.connections`
- `oracle.webcenter.peopleconnections.wall`
- `oracle.webcenter.activitystreaming`

Look in *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal* for more information

42

People Connections Service REST APIs

Oracle WebCenter Portal provides REST APIs to support the People Connections service. You can use the People Connections service REST APIs to perform the following actions:

- View a stream of your connections' activities in an activity stream
- Browse a profile or connections list
- Update profile status messages
- Create and delete connection lists
- Add and remove members from connection lists
- Create, read, and delete messages
- Read and delete feedback

This chapter describes the REST APIs associated with the People Connections service. It includes the following sections:

- Section 42.1, "Activity Stream REST APIs"
- Section 42.2, "Connections and Profile REST APIs"
- Section 42.3, "Feedback REST APIs"
- Section 42.4, "Message Board REST APIs"
- Section 42.5, "Creating an Invitation"

> **See Also:** For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

42.1 Activity Stream REST APIs

Use the Activity Stream REST APIs to browse user application activities in an activity stream. This section provides information about the REST APIs to use to perform this action. It includes the following subsections:

- Section 42.1.1, "Activity Stream Entry Point"
- Section 42.1.2, "Activity Stream Resource Type Taxonomy"
- Section 42.1.3, "Activity Stream Security Considerations"
- Section 42.1.4, "Activity Stream Resource Types"

Note: Because REST APIs can be configured in many different ways, it's possible that not all of a user's activities will be returned, allowing the REST client to customize how the Activity Stream behaves separately from the Webcenter Portal application.

42.1.1 Activity Stream Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. For example, to find the entry point for the Activity Stream feature of the People Connections service, find the link elements with a `resourceType` of:

```
urn:oracle:webcenter:activities:stream
```

The corresponding `href` or `template` element provides the URI entry point, which retrieves application activities for the current user from the Activity Stream. The client sends HTTP requests to this entry point to work with the Activity Stream feature of the People Connections service.

See Also: For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

42.1.2 Activity Stream Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 42.1.4, "Activity Stream Resource Types."

The resource type taxonomy for the Activity Stream feature of the People Connections service is:

```
urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:activity
```

42.1.3 Activity Stream Security Considerations

You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

See Also: For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

42.1.4 Activity Stream Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- "urn:oracle:webcenter:activities:stream"
- "urn:oracle:webcenter:activities:activity"

urn:oracle:webcenter:activities:stream

The `stream` response contains URIs for use in retrieving activities from the Activity Stream.

You can retrieve the activities from a user's stream or activities from a user's connections' streams. To have even greater control over which activities to retrieve, use the activity stream query filter. With the query filter, you can:

- Specify the user to query
- Include the user's connections' activities in the results
- Include activities from spaces, including the Home space, in the results
- Restrict the results to activities from specific services

The options available to you depend on the path you take to get to the `stream` resource and the `rel` of the link that you use. For example, the activity stream query filter is available only from links with a `rel` attribute of `urn:oracle:webcenter:activities:stream`. If you access the activity stream query filter from the `person` resource, the `personGuid` parameter is prefilled.

Table 42–1 shows the activities returned depending on the `rel` element of the link.

Table 42–1 Activities Returned by stream

rel	Returns
`urn:oracle:webcenter:activities:stream:person`	Activities from the user's stream (*`GUID`*`/@self`)[1]
`urn:oracle:webcenter:activities:stream:connections`	Activities from the user's connections' streams (*`GUID`*`/@connections`)[1]
`urn:oracle:webcenter:activities:stream`	Activities determined by the activity stream query filter
`urn:oracle:webcenter:activities:stream:space`	Activities from the space activity stream
`urn:oracle:webcenter:activities:stream:list`	Activities from the space activity list

[1] *`GUID`* can be any valid user GUID or `@me`.

Navigation Paths to stream

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceIndex
   stream (rel="urn:oracle:webcenter:activities:stream:person" or
               "urn:oracle:webcenter:activities:stream")

resourceIndex
   person
      stream (rel="urn:oracle:webcenter:activities:stream:person" or
                  "urn:oracle:webcenter:activities:stream:connections" or
                  "urn:oracle:webcenter:activities:stream")

resourceIndex
   person
      list
         stream

resourceIndex
   space
```

```
      stream

stream
   activity
      param
         stream (rel="urn:oracle:webcenter:activities:stream:space")

personReference
   stream
```

Supported Methods for stream

The following method is supported by the `stream` resource:

- `GET`
  - **request—Parameters:**
    - `startIndex`, `itemsPerPage`

      See Also: For information about REST API parameters, such as `startIndex` and `itemsPerPage`, see "Common Request Query Parameters".

      The following additional parameters are available for the query filter URI:

      - `data` – Returns specified data only (for more information, see "Common Request Query Parameters"). For the stream resource, if you specify the constant `'data'` as the data parameter, all of the basic information about the resource is returned except "comments" and "likes" summaries. If you want to return comments or likes, specify the `data` parameter value `'commentsSummary'` or `'likesSummary'`.

      Note: You can specify multiple data values as a comma separated list. For example, `data=data, commentsSummary`.

      - `personGuid`—(Required) Retrieves activities from the stream for the specified user. **Valid values:** any valid user GUID or `@me`.
      - `serviceIds`—Retrieves activities only for the specified services. **Valid values:** An asterisk (*) returns all services. If null or empty, uses the user preference settings for service filters (from the settings link in the top bar).
      - `personal`—Includes the specified user's activities in the Home space. **Valid values:** `true` or `false`. **Default value:** `false`.
      - `connections`—Includes activities from the streams of the specified user's connections. **Valid values:** `true` or `false`. **Default value:** `false`.
      - `groupSpaces`—Includes activities from all spaces of which the specified user is a member. **Valid values:** `true` or `false`. **Default value:** `false`.
      - `connectionListIds` – A comma separated list of connection list names that specifies the connection lists used to show activities.
      - `groupSpaceGuids` – A comma separated list of space GUIDs used to show activities.

* `userGroupSpaceActivities` – Specifies whether or not to show the user's activities in their spaces. **Valid values:** `true` or `false`. **Default value:** `false`.
* `followedObjects` – Specifies whether or not to show all activities for followed objects that both the current user and the specified user follow. **Valid values:** `true` or `false`. **Default value:** `false`.
* `followedObjectsUserActivities` – Specifies whether or not to show the specified user's activities for followed objects that both the current user and the specified user follow. **Valid values:** `true` or `false`. **Default value:** `false`.
* `advancedQuery` – Specifies filters against streamed activities. Create filters for user names, service IDs, and object details, such as a document's display name.

Note: You must plug actual values into the `advancedQuery` parameter. You cannot pass EL expressions directly into the parameter. Typically, the REST API client handles an EL transformation manually and inserts the actual object data value into the REST URL. See the example below, and see also Section 40.3, "What You Should Know About the Activity Stream Advanced Query Option."

For example, the following URI returns activities from the current user's activity stream, for all services that the user has configured in their personal preference settings for service filters. It returns activities from the user's personal space and from the streams of the user's connections:

```
http://host:port/rest/api/activites?personal=true&connections=true&personGu
id=@me&token=token
```

The following URI returns only the user's personal profile and connections activities:

```
http://host:port/rest/api/activities?serviceIds=oracle.webcenter.peopleconn
ections.profile,oracle.webcenter.peopleconnections.connections&personal=
true&personGuid=@me&token=token
```

This next example illustrates how to use the `advancedQuery` parameter. As explained previously, you cannot pass an EL expression to `advancedQuery`. The REST API client must first obtain the actual data object value, and that value can then be passed to `advancedQuery`. For example, to filter activities for a particular space, you can pass the GUID of the space's scope to `advancedQuery`:

```
http://host:port/rest/api/activities?personGuid=@me&advancedQuery=AE.SCOPE_
ID%20%3D%20\%27s8bba98ff_4cbb_40b8_beee_296c916a23ed\%27&ttoken=token
```

where `s8bba98ff_4cbb_40b8_beee_296c916a23ed` is the GUID of the space. Note that the query string must be encoded with the appropriate escape codes. For a list of EL expressions that can be used to obtain values for `advancedQuery`, see Section 40.3, "What You Should Know About the Activity Stream Advanced Query Option."

– **response—Body:** 0 or more activities

Note: Because the `stream` resource includes activity items, the response may also return resource links for the objects referenced in the activity.

Resource Types Linked to From stream

Table 42–2 lists the resource types that the client can link to from the `stream` resource.

Table 42–2 Related Resource Types for stream

rel	resourceType
self urn:oracle:webcenter:activities:stream:person	urn:oracle:webcenter:activities:stream
	urn:oracle:webcenter:activities:activity

urn:oracle:webcenter:activities:activity

The `activity` response contains the data for activities and URIs for use in retrieving all the data you need from an activity that is included in an Activity Stream.

Navigation Paths to activity

This section shows how the client can navigate through the hypermedia to access the `activity` resource:

```
resourceIndex
   stream
      activity

resourceIndex
   person
      stream
         activity

resourceIndex
   space
      stream
         activity
```

Supported Methods for activity

No methods are supported for `activity`. Activities are currently available only from the `stream` resource type.

Resource Types Linked to from activity

Table 42–3 lists the resource types that the client can link to from this resource.

Table 42–3 Related Resource Types for urn:oracle:webcenter:activities:activity

rel	resourceType
self	`urn:oracle:webcenter:activities:activity`
icon	`urn:oracle:webcenter:activities:activity:icon`

Read-only Elements for activity

Table 42–4 lists the read-only elements for the `activity` resource.

See Also: For information about `projection`, see "Common Request Query Parameters".

Note: The activity will also return a link to an activity icon in the links section of the response, if an icon is available. See urn:oracle:webcenter:activities:activity:icon.

Table 42–4 Read-Only Elements for activity

Element	Type	Description
`activityType`	String	Activity type Unique within the service. For example the Discussions service can return the `activityType` `createForum`.
`commentsCount`	String	If you specify `commentsSummary` in the data parameter, then the `commentsCount` parameter will be returned.
`createdDate`	Date (String)[1]	Date the activity was created.
`description`	String	The description of the activity.
`detail`	String	Detail information for the activity, if available. For example, this might return the contents of a message, the file name of a document, and the like. Similar to `detailURL`. Both, either, or neither can be used.
`detailURL`	String	Detail information for the activity, if available. Similar to `detail`. Both, either, or neither can be used.
`displayDescription`	String	A pre-formatted, internationalized description.
`displayMessage`	String	A pre-formatted, internationalized message (does not include template information).
`groupSpace`	groupSpaceReference	Information about the space in which the activity was performed **Note:** This reference is not present for activities that did not happen in a space (for example, activities that happened in the Home space). Also, because creation of a space happens in a personal space, that activity does not have this element either.
`id`	String	Unique ID of the message
`isSummary`	true or false	Indicates whether or not this activity is a summary of other activities.
`likesCount`	String	If you specify `likesSummary` in the data parameter, then the `likesCount` parameter will be returned.
`message`	String	Localized string for this activity This may contain replacement strings within curly braces ({}).

Table 42–4 (Cont.) Read-Only Elements for activity

Element	Type	Description
`permission`	PRIVATE SHARED PUBLIC	Permission level of this activity
`scope`	String	Scope of the activity This might return a space, like the Home space. For example, for a space, it returns a string similar to the following: `s8bba98ff_4cbb_40b8_beee_296c916a23ed`
`serviceId`	String	Unique ID of the service that created the activity **Note:** For a list of WebCenter Portal service IDs, see Table G–7, " Service IDs".
`templateParams`	urn:oracle:webcenter:activities:parameter	A list of `param` elements that capture data related to an activity. A key provided with each `templateParam` <param> element allows you to plug one or more data items into a parameterized activity message. See "Understanding the templateParams Element".
`custom`	Param	The custom parameter includes a `displayName`, `key`, and `type`, and may or may not have a URL.

[1] Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`. The `DATE` data type returns a Java standard date format, for example, `2009-08-21T14:43:11.0013-0700`, with `0700` representing the time zone.

Understanding the templateParams Element

Suppose you want to display the most recent messages for a user named Carl. You want to display information like this: "Carl created the space Customer Feedback". The `templateParams` elements helps you solve this problem.

The `templateParams` element is returned in objects of type `oracle:webcenter:activities:activity`. This element captures a lot of data related to an activity. For example, if a user creates a new space, the `templateParams` element for that activity captures information about the user and about the space. Keys are provided that allow you to perform string substitutions in a parameterized message string related to the activity.

For example, if the user creates a space, the returned activity object contains a `<message>` item that is parameterized like this:

```
<message>{actor[0]} has created the space {object[0]}</message>
```

By parsing the `templateParams` element for the activity, you can find the available keys that allow you to perform string substitutions as well as appropriate data to display, like the name of the user and the activity.

For a detailed example showing how you might code a UI using `templateParams` to display information about an activity, see Section 54.13.2, "Displaying Activity Stream Data."

The templateParams element also provides links to comments, likes, commentsSummary, and likesSummary objects, if they are requested using the data parameter. These links let you query for all the comments or likes for an object or post a new comment or like. The summary links returns the comments or likes count and several recent comments or likes. See also "Understanding Comments and Likes".

The links returned with the templateParams element vary depending on what kind of object is returned, like a user, document, space (GS), or custom object. For more information, see "urn:oracle:webcenter:activities:parameter".

> **Note:** The templateParams element can sometimes contain elements that are not directly referenced in the message.

If a rel link is marked "via," this means it is a link to the underlying REST object–for instance, the document not the parameter. If a rel link is marked "alternate" and type "text/html," it is a link to the HTML page for that object. Space (GS) objects include an activity stream link for spaces. Users have icon and activity stream links. Other objects can have an alt link to the tagged item, as well as the related tagged items list, if tagging is enabled for that object. If a regular object supports comments or likes, it can include the comments/commentsSummary and likes/likesSummary, as explained previously.

Understanding Comments and Likes

Likes and Comments only show up if they are requested using the data parameter. They do not appear by default. Comments are associated with the object referenced by the activity, not the activity itself. For example, if you edit a document, then the comments will be associated with the document, not the edit activity. This means that if you "like" the document, then the comments from editing it will also show up.

The links associated with comments and likes are:

- comments – The comments link lets you query for all the comments for an object. This link also lets you POST a new comment. For a comments POST, the text field is required.
- commentsSummary – The commentsSummary link returns the comments count and several recent comments.
- likes – The likes link lets you query for all the likes for an object. This link also lets you POST a new like. There are no required fields for a likes POST.
- likesSummary – The likesSummary link returns the likes count and the current user's like (if available).

> **Note:** Like objects themselves only support DELETE, not GET. The likesCount and commentsCount items return the number of likes or comments for the current object.

urn:oracle:webcenter:activities:parameter

The templateParams element returns a set of param elements. The param elements return data specific to the type of activity object returned. The possible types of param elements include:

- user – Returns the displayName, guid, id, key, primaryId, and type.

- `document` – Returns the `displayName`, `iconUrl`, `id`, `key`, `primaryId`, and `type`.
- `custom` – Returns `displayName`, `key`, and `type`, and may or may not have a URL.

A param can also be a variable reference or key of the general form `{prefix[index].variable}`.

urn:oracle:webcenter:activities:activity:icon

Use this resource type to get the icon of the activity, if available (`GET`).

Navigation Paths to activities:activity:icon

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   activities
      activity
         icon
```

Supported Methods for icon

The following methods are supported by this resource:

- `GET` – Returns the icon used for the named activity.

Resource Types Linked to from icon

Table 42–5 lists the resource types that the client can link to from this resource.

Table 42–5 Related Resource Types for urn:oracle:webcenter:activities:activity:icon

rel	resourceType
self	`urn:oracle:webcenter:activities:activity:icon`
urn:oracle:webcenter:parent	`urn:oracle:webcenter:activities:activity`

42.2 Connections and Profile REST APIs

Use the Connections and Profile REST APIs to browse a profile or a connections list, manage connections lists, add or remove connections, send invitations to connect, and update a profile status message.

This section provides information about the REST APIs to use to perform these actions. It includes the following subsections:

- Section 42.2.1, "Connections and Profile Entry Point"
- Section 42.2.2, "Connections and Profile Resource Type Taxonomy"
- Section 42.2.3, "Connections and Profile Security Considerations"
- Section 42.2.4, "Connections and Profile Resource Types"

42.2.1 Connections and Profile Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own link element. For example, to find the entry points for Connections and Profile

features of the People Connections service, find the link elements with a `resourceType` of:

`urn:oracle:webcenter:people` (returns the current user profile)

`urn:oracle:webcenter:people:person` (enables you to query for a user)

`urn:oracle:webcenter:people:invitations` (returns invitations sent or received by the current user)

Note: The `people:person` and `people:invitations` resource types have a `template` but not an `href`.

The corresponding `href` or `template` element provides the URI entry point, which returns a list of people (`people`) or an individual user (`people:person`). The client sends HTTP requests to this entry point to work with the Connections and Profile features of the People Connections service.

See Also: For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

42.2.2 Connections and Profile Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 42.2.4, "Connections and Profile Resource Types."

The resource type taxonomy for the Connections and Profile features in the People Connections service is:

```
urn:oracle:webcenter:people
   urn:oracle:webcenter:people:person
   urn:oracle:webcenter:people:icon
   urn:oracle:webcenter:people:person:list
      urn:oracle:webcenter:people:person:listNames
      urn:oracle:webcenter:people:person:listName
      urn:oracle:webcenter:people:person:list:member
      urn:oracle:webcenter:people:person:status
   urn:oracle:webcenter:people:invitations
      urn:oracle:webcenter:people:invitation
```

42.2.3 Connections and Profile Security Considerations

You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

See Also: For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

42.2.4 Connections and Profile Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- "urn:oracle:webcenter:people"
- "urn:oracle:webcenter:people:person"
- "urn:oracle:webcenter:people:person:list"
- "urn:oracle:webcenter:people:person:listNames"
- "urn:oracle:webcenter:people:person:listName"
- "urn:oracle:webcenter:people:person:list:member"
- "urn:oracle:webcenter:people:person:status"
- "urn:oracle:webcenter:people:invitations"
- "urn:oracle:webcenter:people:invitation"

urn:oracle:webcenter:people

The `people` response contains URIs for use in retrieving the profile of one or more users.

Navigation Paths to people

This section shows how the client can navigate through the hypermedia to access the `people` resource:

```
resourceIndex
   people
```

Supported Methods for people

The following method is supported by the `people` resource:

- `GET`
  - **request—Parameters:**
    - `startIndex` – See "Common Request Query Parameters."
    - `itemsPerPage` – See "Common Request Query Parameters."
    - `projection` – See "Common Request Query Parameters."
    - `data` – The data parameter is a comma separated list that controls which data will be returned in the response. The predefined set `basic` is equivalent to `data=guid,id,displayName`. The predefined set `data` is a standard set that returns all the data for the user, but does not include `status`, `manager`, `reportees`, or `photos`. The full list of possible values includes the predefined sets `basic` and `data`, as well as the following individual data values: `guid`, `id`, `displayName`, `birthday`, `language`, `timeZone`, `name`, `addresses`, `organizations`, `workEmail`, `phoneNumbers`, `manager`, `reportees`, `photos`, and `status`.

      If you specify the constant `'data'` as the `data` parameter, all the basic information will be returned for the resource. If both the `projection` and `data` query string parameters are present, the `data` parameter will be used to determine which data to return.

      The `data` parameter can also take any of the following values comma-separated values to return the corresponding data: `guid`, `id`, `displayName`, `birthday`, `language`, `timeZone`, `name`, `addresses`, `organizations`,

workEmail, phoneNumbers, manager, reportees, photos, and/or status.

Note: The data parameter can accept a predefined set, a collection of values, or a mix of sets and values. For example, to get the basic data plus the user's birthday, you can specify data=basic,birthday.

* links – The links parameter is a comma-separated list that controls which links will be returned in the response. This parameter can accept predefined sets, individual data values, or a combination of predefined sets and individual data values. The predefined sets are basic, data, activitiesSet, connectionsSet, and feedbackSet. These predefined sets are described in "Predefined Sets for the links Parameter".

  The individual values for the links parameter are: person, profile, icon, status, messageBoard, activities, personActivities, connectionActivities, connections, listNames, invitation, givenFeedback, receivedFeedback, userGivenFeedback, manager, reportees, member.

  If both the projection and links query string parameters are present, the links parameter will be used to determine which links to return.

– **response—Body:** One or more person objects.

Predefined Sets for the links Parameter

The following items are predefined sets that can be returned in the links parameter. For example, if you specify links=basic, it is the equivalent of specifying data=person,profile,icon. You can also specify additional parameters as needed. For example, you could specify data=basic,birthday.

Note: Links that are not currently available will not be returned even if they are specified in the links parameter.

- basic – A standard set that returns the basic information for the profile and includes person, profile, and icon.
- data – A standard set that returns all the basic links for the response plus the connections list, status, and activity stream template.
- activitiesSet – Includes activities, personActivities, and connectionActivities.
- connectionsSet – Includes connections, listNames, and invitation.
- feedbackSet – Includes givenFeedback, receivedFeedback, and userGivenFeedback.

Resource Types Linked to from people

Table 42–6 lists the resource types that the client can link to from the people resource.

Table 42–6 Related Resource Types for people

rel	resourceType
urn:oracle:webcenter:people:icon	urn:oracle:webcenter:people:person

Table 42–6 (Cont.) Related Resource Types for people

rel	resourceType
urn:oracle:webcenter:people:person:list:connections	urn:oracle:webcenter:people:person:list
urn:oracle:webcenter:activities:stream:person	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream:connections	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:feedback:all-received	urn:oracle:webcenter:feedback
urn:oracle:webcenter:feedback:all-given	urn:oracle:webcenter:feedback
self	urn:oracle:webcenter:people:person:status
urn:oracle:webcenter:people:person:list:list	urn:oracle:webcenter:people:person:list
self	urn:oracle:webcenter:people:person:listName
urn:oracle:webcenter:activities:stream:list	urn:oracle:webcenter:activities:stream

urn:oracle:webcenter:people:icon

Use this resource type to get the icon used for the named profile (`GET`).

Navigation Paths to people:icon

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   people
      icon
```

Supported Methods for icon

The following methods are supported by this resource:

- `GET` – Returns the icon used for the named profile.

> **Note:** This resource includes a template that lets you choose the size of the profile icon to use. The `size` template parameter can be set to `small`, `medium`, or `large`.

Resource Types Linked to from icon

Table 42–7 lists the resource types that the client can link to from this resource.

Table 42–7 Related Resource Types for urn:oracle:webcenter:people:icon

rel	resourceType
self	`urn:oracle:webcenter:people:icon`
urn:oracle:webcenter:parent	`urn:oracle:webcenter:people`

urn:oracle:webcenter:people:person

The `person` response contains profile data and the URIs for use in retrieving a user profile.

Navigation Paths to person

This section shows how the client can navigate through the hypermedia to access the `person` resource:

```
resourceIndex
   people

resourceIndex
   people
      person

resourceIndex
   person

author
   person

resourceIndex
   activities:stream
      person
```

Supported Methods for person

The following method is supported by the `person` resource:

- `GET`
  - **request—Parameters:** `q`

    To retrieve a specified person, the format of `q` is:

    ```
    q=[loginid:equals:username]
    Or
    q=[guid:equals:guid]
    Or
    q=[email:equals:email]
    ```

    Note: The parameter `q` is only on the `resourceIndex` template for `person`.

  - **response—Body:** `message`

    See Also: For information about the response `message`, see "Read-only Elements for person".

Read-only Elements for person

Table 42–8 lists the read-only elements for the `person` resource.

Note: The elements present in a `person` response depend on how the user repository is configured and the elements it supports. Additionally, several of the pieces of data represented in Table 42–8, such as `address`, `emails`, `photos`, `phoneNumbers`, and `organization`, can have multiple instances.

See Also: Some of the elements listed in Table 42–8 can be returned in predefined sets described in "Predefined Sets for the links Parameter".

Table 42–8 Read-Only Elements for person

Element	Type	Description
`guid`	String	Unique GUID of the person
`id`	String	Unique login ID of the person (that is, the user name, for example, pat_coi)
`displayName`	String	Display name of the person (the user's name, for example, Pat Coi). This may have the same value as `id`, depending on the repository configuration.
`birthday`	Date (String)[1]	Birth date of the person
`connected`	Boolean (String)1	Whether or not this person is connected to the current user
`language`	String	Preferred language of the person
`timeZone`	String	Time zone of the person
`name`	`name`	Name information for the person `name` is a portable contact type. For more information, see Section 54.10.2.1, "name Portable Contact Type."
`address`	`address`	Address information for the person address is a portable contact type. For more information, see Section 54.10.2.2, "address Portable Contact Type."
`emails`	`value`	Emails for the person `emails` is derived from the `value` portable contact type. For more information, see Section 54.10.2.4, "value Portable Contact Type."
`photos`	`value`	Profile photos for the person `photos` is derived from the `value` portable contact type. For more information, see Section 54.10.2.4, "value Portable Contact Type."
`phoneNumbers`	`value`	Phone numbers for the person `phoneNumber` is derived from the `value` portable contact type. For more information, see Section 54.10.2.4, "value Portable Contact Type."
`organizations`	`organization`	Organization information for the person organization is a portable contact type. For more information, see Section 54.10.2.3, "organization Portable Contact Type."

Table 42–8 (Cont.) Read-Only Elements for person

Element	Type	Description
`manager`	personReference	Manager of this person
`reportees`	personReference	Direct reports of this person
`status`	urn:oracle:webcenter:people:person:status	Person's profile status message

[1] Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

Resource Types Linked to from person

Table 42–9 lists the resource types that the client can link to from the `person` resource.

Table 42–9 Related Resource Types for person

rel	resourceType
self	urn:oracle:webcenter:people:person
alternate	urn:oracle:webcenter:spaces:profile (HTML)
urn:oracle:webcenter:people:person:list:connections	urn:oracle:webcenter:people:person:list
	urn:oracle:webcenter:people:person:listNames
	urn:oracle:webcenter:people:person:status
urn:oracle:webcenter:activities:stream:person	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream:connections	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream	urn:oracle:webcenter:activities:stream
	urn:oracle:webcenter:messageBoard
urn:oracle:webcenter:feedback:all-given	urn:oracle:webcenter:feedback
urn:oracle:webcenter:feedback:all-received	urn:oracle:webcenter:feedback

urn:oracle:webcenter:people:person:list

The `list` response contains URIs for use in retrieving all the profiles on a connections list (`GET`), inviting a user to be a connection or adding a connection to a connections list (`POST`), and removing a connections list (`DELETE`).

Navigation Paths to list

This section shows how the client can navigate through the hypermedia to access the `list` resource:

```
resourceIndex
   person
      listNames
         list

resourceIndex
   person
      list (rel="urn:oracle:webcenter:people:person:list:connections")
```

Supported Methods for list

The following methods are supported by the `list` resource:

- `GET`
  - **request—Parameters:** `startIndex`, `itemsPerPage`, `projection`

See Also: For information about REST API parameters, such as `startIndex` and `itemsPerPage`, see "Common Request Query Parameters".

  - **response—Body:** 0 or more `person` items

- `POST`
  - **request—Body:** `member`
  - **response—Body:** `member`

    See Also: For information about `member` in the request and response elements, see "urn:oracle:webcenter:people:person:list:member".

- `DELETE`
  - **request**

Resource Types Linked to from list

Table 42–10 lists the resource types that the client can link to from the `list` resource.

Table 42–10 Related Resource Types for list

rel	resourceType
self urn:oracle:webcenter:people:person:list[1]	urn:oracle:webcenter:people:person:list
urn:oracle:webcenter:activities:stream	urn:oracle:webcenter:activities:stream

[1] `self rel` currently includes "`urn:oracle:webcenter:people:person:list:list`" instead of the correct "`urn:oracle:webcenter:people:person:list`". For the `@connections` default list, it currently includes "`urn:oracle:webcenter:people:person:list:connections`".

urn:oracle:webcenter:people:person:listNames

The `listNames` response contains the names of existing connections lists as well as the URIs for use in retrieving the lists (GET) and creating connections lists (`POST`).

Navigation Paths to listNames

This section shows how the client can navigate through the hypermedia to access the `listNames` resource:

```
resourceIndex
   person
      listNames
```

Supported Methods for listNames

The following methods are supported by the `listNames` resource:

- `GET`
  - **request**
  - **response**: **Body:** 0 or more `listName` items
- `POST`
  - **request—Body:** `listName`
  - **response—Body:** `listName`

See Also: For information about `listName`, see "urn:oracle:webcenter:people:person:listName".

Resource Types Linked to from listNames

Table 42–11 lists the resource types that the client can link to from the `listNames` resource.

Table 42–11 Related Resource Types for listNames

rel	resourceType
self	urn:oracle:webcenter:people:person:listNames

urn:oracle:webcenter:people:person:listName

The `listName` response contains the names of connections lists and URIs for use in accessing the connections lists.

Navigation Paths to listName

This section shows how the client can navigate through the hypermedia to access the `listName` resource:

```
resourceIndex
   person
      listNames
         listName
```

Writable Elements for listName

Table 42–12 lists the writable elements for the `listName` resource.

Table 42–12 Writable Elements for listName

Element	Type	Required	Constraints	Description
`name`	String	Yes	1 or more characters	A single list name

Resource Types Linked to from listName

Table 42–13 lists the resource types that the client can link to from the `listName` resource.

Table 42–13 Related Resource Types for listName

rel	resourceType
self urn:oracle:webcenter:people:person:list	urn:oracle:webcenter:people:person:list
urn:oracle:webcenter:activities:stream:list	urn:oracle:webcenter:activities:stream

urn:oracle:webcenter:people:person:list:member

The `member` response contains URIs for use in deleting a connection from a connections list.

Navigation Paths to member

This section shows how the client can navigate through the hypermedia to access the `member` resource:

```
resourceIndex
```

```
person
   list
      member
```

Supported Methods for member

The following method is supported by the `member` resource:

- `DELETE`
  - **request**

Writable Elements for member

Table 42–14 lists the writable elements for the `member` resource. Writable elements for `member` are used when you add a connection to a list or invite a user to be a connection. The `member` resource itself is for deleting connections, and does not use writable elements.

Table 42–14 Writable Elements for member

Element	Type	Required	Constraints	Description
`guid`	String	Yes	1 or more characters	GUID of the user
`message`	String	No	0 or more characters	Invitation message Use this only when inviting users to be connections (that is POSTing to the `@connections` list, not to user-created connections lists).

urn:oracle:webcenter:people:person:status

The `status` response contains URIs for use in retrieving (`GET`) and updating (`PUT`) the profile status message of a specified user.

Navigation Paths to status

This section shows how the client can navigate through the hypermedia to access the `status` resource:

```
resourceIndex
   people
      person
         status
```

Supported Methods for status

The following methods are supported by the `status` resource:

- `GET`
  - **request**
  - **response—Body:** `status`
- `PUT`
  - **request—Body:** `status`
  - **response—Body:** `status`

Writable Elements for status

Table 42–15 lists the writable elements for the `status` resource.

Table 42–15 Writable Elements for status

Element	Type	Required	Constraints	Description
`note`	String	Yes	1 or more characters	Content of status message

Resource Types Linked to from status

Table 42–16 lists the resource types that the client can link to from the `status` resource.

Table 42–16 Related Resource Types for status

rel	resourceType
self	urn:oracle:webcenter:people:person:status

urn:oracle:webcenter:people:invitations

The `invitations` response contains URIs for use in retrieving invitations (`GET`). You can also send an invitation (`POST`) to another user.

Navigation Paths to invitations

This section shows how the client can navigate through the hypermedia to access the `invitations` resource:

```
resourceIndex
   invitations
```

If you are not already connected to a user, you can also navigate to the invitations resource from that user's profile in order to invite them to connect. This path is only used for POSTing.

```
resourceIndex
   person
      invitations
```

Supported Methods for invitations

The following methods are supported by the `invitations` resource:

- `GET`
  - **Request—Parameters:** `q`

    To retrieve invitations sent to the current user, the format of `q` is:

    ```
    q=[invitee:equals:@me]
    ```

    To retrieve invitations sent from the current user, the format of `q` is:

    ```
    q=[invitor:equals:@me]
    ```

  - **Response—Body:** 1 or more invitations
- `POST`
  - **Request—Body:** `invitation`
  - **Response—Body:** `invitation`

Writable Elements for invitations

Table 42–17 lists the writable elements for the `invitations` resource.

Table 42–17 Writable Elements for invitations

Element	Type	Required	Constraints	Description
`message`	String	No	1 or more characters	Message attached to the invitation

Resource Types Linked to from invitations

Table 42–18 lists the resource types that the client can link to from the `invitations` resource:

Table 42–18 Related Resource Types for invitations

rel	resourceType
self	urn:oracle:webcenter:people:invitation

urn:oracle:webcenter:people:invitation

The `invitation` response contains URIs for use in deleting (`DELETE`) invitations you have sent, or deleting (`DELETE`) or updating (`PUT`) invitations you have received.

Navigation Paths to invitation

This section shows how the client can navigate through the hypermedia to access the `invitation` resource:

```
resourceIndex
   invitations
      invitation
```

Supported Methods for invitation

The following methods are supported by the `invitation` resource:

- `PUT`
  - **Request—Body:** `invitation`
  - **Response—Body:** `invitation`
- `DELETE`
  - **Request**

Writable Elements for invitation

Table 42–19 lists the writable elements for the `invitation` resource.

Table 42–19 Writable Elements for invitation

Element	Type	Required	Constraints	Description
`status`	String	Yes (PUT)	ACCEPTED IGNORED	The status of the invitation. **Note:** When you accept or ignore an invitation, it is removed from your list of invitations.

Read-only Elements for invitation

Table 42–20 lists the read-only elements for the `invitation` resource.

Table 42–20 Read-only Elements for invitation

Element	Type	Description
`id`	String	Unique ID of the invitation
`invitee`	personReference	User to whom the invitation is sent
`invitor`	personReference	User from whom the invitation is sent
`sentDate`	Date (String)[1]	Date the invitation was sent

[1] Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

Resource Types Linked to from invitation

Table 42–21 lists the resource types that the client can link to from the `invitation` response.

Table 42–21 Related Resource Types for invitations

rel	resourceType
self	urn:oracle:webcenter:people:invitation

42.3 Feedback REST APIs

Use the Feedback REST APIs to read and delete feedback. This section provides information about the REST APIs to use to perform these actions. It includes the following subsections:

- Section 42.3.1, "Feedback Entry Point"
- Section 42.3.2, "Feedback Resource Type Taxonomy"
- Section 42.3.3, "Feedback Security Considerations"
- Section 42.3.4, "Feedback Resource Types"

42.3.1 Feedback Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own `link` element. To find the entry points for the Feedback feature of the People Connections service, find the `link` elements with a `resourceType` of:

```
urn:oracle:webcenter:feedback
```

The corresponding `href` or `template` element provides the URI entry point, which returns all received feedback for the current user. The client sends HTTP requests to this entry point to work with the Feedback feature of the People Connections service.

> **See Also:** For more information about the Resource Index, see Section 54.5.1, "The Resource Index."
>
> For more information about resource types, see Section 54.5.2.1, "Resource Type."

42.3.2 Feedback Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 42.3.4, "Feedback Resource Types."

the resource type taxonomy for the Feedback feature of the People Connections service is:

```
urn:oracle:webcenter:feedback
   urn:oracle:webcenter:feedback:message
```

42.3.3 Feedback Security Considerations

You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

> **See Also:** For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

42.3.4 Feedback Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- "urn:oracle:webcenter:feedback"
- "urn:oracle:webcenter:feedback:message"

urn:oracle:webcenter:feedback

The `feedback` response contains URIs for use in reading Feedback messages.

Navigation Paths to feedback

This section shows how the client can navigate through the hypermedia to access the `feedback` resource:

```
resourceIndex
   feedback

resourceIndex
   person
      feedback
```

Supported Methods for feedback

The following methods are supported by the `feedback` resource:

- `GET`
  - **request—Parameters:** `startIndex`, `itemsPerPage`

    > **See Also:** For information about REST API parameters, such as `startIndex` and `itemsPerPage`, see "Common Request Query Parameters".

  - **response—Body:** `message`

See Also: For information about `message`, see "urn:oracle:webcenter:feedback:message".

- `POST` – If permitted, lets you add feedback for a target user. This method is only available if the current user is connected to and has permission to add feedback for the target user.
  - **request - body**: feedback

```
<message resourceType="urn:oracle:webcenter:feedback:message">
    <body>test from REST API</body>
    <receivedUser>
    <guid>4F16DD80393611DFBF895F177662C511</guid>
    </receivedUser>
</message>
```

Resource Types Linked to from feedback

Table 42–22 lists the resource types that the client can link to from the `feedback` resource.

Table 42–22 Related Resource Types for feedback

rel	resourceType
self urn:oracle:webcenter:feedback:all-received	urn:oracle:webcenter:feedback
self urn:oracle:webcenter:feedback:all-given	urn:oracle:webcenter:feedback
	urn:oracle:webcenter:feedback:message

urn:oracle:webcenter:feedback:message

The `message` response contains the feedback message data and URIs for use in deleting a Feedback message.

Navigation Paths to message

This section shows how the client can navigate through the hypermedia to access the `message` resource:

```
resourceIndex
   feedback
      message

resourceIndex
   person
      feedback
         message
```

Supported Methods from message

The following method is supported by the `message` resource:

- `DELETE`
  - **request**

Read-only Elements for message

Table 42–23 lists the read-only elements for the `message` resource.

Table 42–23 Read-only Elements for message

Element	Type	Description
`body`	String	Message content
`id`	String	Unique ID of the message
`author`	personReference	User who created the message
`created`	Date (String)[1]	Date the message was created
receivedUser	**personReference**	A person reference to the user who received the feedback

[1] Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

Resource Types Linked to from feedback

Table 42–24 lists the resource types that the client can link to from the `feedback` resource.

Table 42–24 Related Resource Types for message

rel	resourceType
self	urn:oracle:webcenter:feedback:message

42.4 Message Board REST APIs

Use the Message Board REST APIs to post, read, and delete messages to a user's or a space message board.

This section provides information about the REST APIs to use to perform these actions. It includes the following subsections:

- Section 42.4.1, "Message Board Entry Point"
- Section 42.4.2, "Message Board Resource Type Taxonomy"
- Section 42.4.3, "Message Board Security Considerations"
- Section 42.4.4, "Message Board Resource Types"

42.4.1 Message Board Entry Point

Each REST service has a `link` element within the Resource Index that provides the entry point for that service. For the People Connections service, each feature has its own `link` element. To find the entry points for the Message Board feature of the People Connections service, find the `link` elements with a `resourceType` of:

```
urn:oracle:webcenter:messageBoard
```

> **Note:** As well as an entry point from the Resource Index, to navigate to an individual user's message board, the Message Board feature also has an entry point from a space response for the space message board.

The corresponding `href` or `template` element provides the URI entry point, which returns the Message Board for the current user. The client sends HTTP requests to this entry point to work with the Message Board feature of the People Connections service.

See Also: For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

42.4.2 Message Board Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 42.4.4, "Message Board Resource Types."

The resource type taxonomy for the Message Board feature of the People Connections service is:

```
urn:oracle:webcenter:messageBoard
   urn:oracle:webcenter:messageBoard:message
```

42.4.3 Message Board Security Considerations

You must be logged in to the REST service to access any of the People Connections REST APIs. After that, the underlying service handles permission checking and the like.

See Also: For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

42.4.4 Message Board Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- "urn:oracle:webcenter:messageBoard"
- "urn:oracle:webcenter:messageBoard:message"

urn:oracle:webcenter:messageBoard

The `messageBoard` response contains URIs for use in reading (`GET`) and posting (`POST`) space and individual user Message Board messages.

Navigation Paths to messageBoard

This section shows how the client can navigate through the hypermedia to access the `messageBoard` resource:

```
resourceIndex
   messageBoard

resourceIndex
   person
      messageBoard

resourceIndex
   spaces
      space
         messageBoard
```

Supported Methods for messageBoard

The following methods are supported by the `messageBoard` resource:

- `GET`
  - **request—Parameters:** `startIndex`, `itemsPerPage`

    See Also: For information about REST API parameters, such as `startIndex` and `itemsPerPage`, see "Common Request Query Parameters".

  - **response—Body:** message

    Note: The REST `GET` command for reading (retrieving) messages retrieves all shown messages. Messages that are hidden through the application user interface are not retrieved. For information about hiding and showing messages, see "Working with Message Board Task Flows," in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

- `POST`
  - **request—Body:** `message`

    The POST for messages supports including a link URL in the message.

    See Also: For information about message, see "urn:oracle:webcenter:messageBoard:message".

Read-only Elements for messageBoard

Table 42–25 lists the read-only elements for the `messageBoard` resource.

Table 42–25 Read-only Elements for message

Element	Type	Description
`messageType`	String	Returns `link` if the message has a link. Otherwise, returns `null`.
`link`	String	Contains link data for messages with links: `name`, `url`, `icon`, `description`, `mimeType`, `objectId`, `objectType`, `serviceId`.

Resource Types Linked to from messageBoard

Table 42–26 lists the resource types that the client can link to from the `messageBoard` resource.

Table 42–26 Related Resource Types for messageBoard

rel	resourceType
self	urn:oracle:webcenter:messageBoard
	urn:oracle:webcenter:messageBoard:message

urn:oracle:webcenter:messageBoard:message

The `message` response contains the Message Board message data and URIs for use in reading (`GET`), revising (`PUT`), and deleting (`DELETE`) a space or individual user Message Board message.

Navigation Paths to message

This section shows how the client can navigate through the hypermedia to access the `message` resource:

```
resourceIndex
   messageBoard
      message

resourceIndex
   person
      messageBoard
         message

resourceIndex
   spaces
      space
         messageBoard
            message
```

Supported Methods for message

The following methods are supported by the `message` resource:

- `GET`
  - **request**
  - **response—Body:** `message`
- `PUT`
  - **request—Body:** `message`
  - **response—Body:** `message`
- `DELETE`
  - **request**

Writable Elements for message

Table 42–27 lists the writable elements for the `message` resource.

Table 42–27 Writable Elements for message

Element	Type	Required	Constraints	Description
`body`	String	Yes	1 or more characters	message content

Read-only Elements for message

Table 42–28 lists the read-only elements for the `message` resource.

Table 42–28 Read-only Elements for message

Element	Type	Description
`id`	String	Unique ID of the message
`author`	personReference	User who created the message
`created`	Date (String)[1]	Date the message was created

[1] Data types, such as `DATE` and `BOOLEAN`, are stored in the API as `STRING`.

Resource Types Linked to from message

Table 42–29 lists the resource types that the client can link to from the `message` resource.

Table 42–29 Related Resource Types for message

rel	resourceType
self	urn:oracle:webcenter:messageBoard:message

42.5 Creating an Invitation

This section illustrates how to invite another user to join your connections list using the People Connections Service REST API. After the invitation is made, the invitee is given the option to accept or not. This example also illustrates how to delete an invitation.

This section includes the following subsections:

- Section 42.5.1, "Creating an Invitation"
- Section 42.5.2, "Deleting an Invitation"

42.5.1 Creating an Invitation

1. The first step, as always with the REST APIs, is to retrieve the resource index:

   ```
   GET http://<host:port>/rest/api/resourceIndex
   ```

2. Next, find the person to whom you want to connect. To do this:

   1. In the resource index listing, scan for the link with the following resource type:

      ```
      urn:oracle:webcenter:people:person
      ```

   2. Execute a search for the user who you wish to invite. In this example, the user is named Monty.

      ```
      GET http://<host:port>/rest/api/people?q=loginid:equals:monty&utoken=ASDF
      ```

   3. Locate Monty's GUID in the response, as shown in Figure 42–1.

Figure 42–1 Response With User's GUID

```
</links>
<connected>false</connected>
<displayName>monty</displayName>
<guid>1AE5AF102E2611E09F062B573E287934</guid>
<id>monty</id>
−<name>
  <formatted>monty</formatted>
</name>
```

   4. Save the GUID so that you can use it to connect to the user.

3. Now that you have the invitee's GUID, you must find the resource to which you would like to add him. In this case, you want to add him to your own `connections` list.

   To find your `connections` list, first, scan the People Connections documentation (this chapter) for "`connections`." You will discover in Table 42–9, " Related

Resource Types for person" that `connections` are linked to from the `person` resource. For convenience, this table is shown in Figure 42–2.

Figure 42–2 Finding the connections Link

rel	resourceType
self	urn:oracle:webcenter:people:person
alternate	urn:oracle:webcenter:spaces:profile (HTML)
urn:oracle:webcenter:people:person:list:connections	urn:oracle:webcenter:people:person:list
	urn:oracle:webcenter:people:person:listNames
	urn:oracle:webcenter:people:person:status
urn:oracle:webcenter:activities:stream:person	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream:connections	urn:oracle:webcenter:activities:stream
urn:oracle:webcenter:activities:stream	urn:oracle:webcenter:activities:stream
	urn:oracle:webcenter:messageBoard
urn:oracle:webcenter:feedback:all-given	urn:oracle:webcenter:feedback
urn:oracle:webcenter:feedback:all-received	urn:oracle:webcenter:feedback

Now that you know that `connections` are linked to from the `person` resource, you need to find the `person` resource. As the URN indicates, you get to the `person` resource from the `people` resource, as described in the following steps.

4. Return to the resource index (or use a cached version of the `resourceIndex` from your previous visit):

   ```
   GET http://<host:port>/rest/api/resourceIndex
   ```

5. Scan for the `people` resource, and you will find:

   ```
   urn:oracle:webcenter:people
   ```

6. Use the link in the `people` resource to access *your* lists:

   ```
   GET http://<host:port>/rest/api/people/@me/lists/@self?utoken=ASDF
   ```

7. Scan the returned links for the `resourceType` and `rel` listed in Table 42–9:

   ```
   resourceType="urn:oracle:webcenter:people:person:list
   rel="urn:oracle:webcenter:people:person:list:connections
   ```

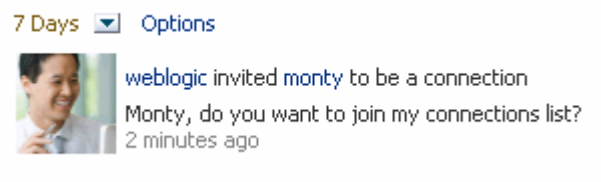
8. Make the invitation by using the URI to your `connections` list to execute a `POST`:

   ```
   POST http://<host:port>/rest/api/people/@me/lists/@connections?utoken=ASDF

   Headers --  Accept:application/json, Content-Type:application/json
   Body -- {"message":"Monty, do you want to join my connections
   list?","guid":"1AE5AF102E2611E09F062B573E287934"}
   ```

9. Now, if you log in to Monty's account and you can see the invitation has been added, as shown in Figure 42–3.

Figure 42–3 Activity Stream Showing Invitation



42.5.2 Deleting an Invitation

After you initiate an invitation, you can view the invitation from your account by specifying `invitor:equals:@me`. For example:

```
GET
http://<host:port>/rest/api/people/invitations?q=invitor:equals:@me&utoken=ASDF
```

Each invitation element listed in the response contains a link that supports the DELETE operation and that looks something like this:

```
<links>
<link resourceType="urn:oracle:webcenter:people:invitation"
rel="self"
href="http://host_name:port_name/rest/api/people/invitations/
     e9073cdb-56ab-423d-8b1f-1220c802bdd4?
     utoken="FN0SEFwX42OCntwtx9a1dSbhqocO_w**"
     capabilities="urn:oracle:webcenter:delete"/>
</links>
```

The invitee can also delete an invitation from his or her own account. The invitee can get a list of his or her invitations by specifying:

```
GET
http://<host:port>/rest/api/people/invitations?q=invitee:equals:@me&utoken=ASDF
```

Note that the response from a DELETE is simply a status code of 204.

Part VII

Helping Users Find Content

Part VII contains the following chapters:

- Chapter 43, "Integrating the Links Service"
- Chapter 44, "Integrating the Tags Service"
- Chapter 45, "Integrating the Search Service"
- Chapter 46, "Integrating the Activity Graph Service"

43 Integrating the Links Service

This chapter explains how to integrate the Links service in a WebCenter Portal: Framework application at design time.

For more information about managing and including links, see:

- "Setting up the WebCenter Portal Repository" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Links Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 43.1, "Introduction to the Links Service"
- Section 43.2, "Basic Configuration for the Links Service"
- Section 43.3, "Advanced Information for the Links Service"

43.1 Introduction to the Links Service

This section provides overview information about the Links service features and requirements. It includes the following subsections:

- Section 43.1.1, "Understanding the Links Service"
- Section 43.1.2, "Requirements for the Links Service"
- Section 43.1.3, "What Happens at Runtime"

43.1.1 Understanding the Links Service

The Links service provides a way to view, access, and associate related information. For example, in a list of project assignments, you can link to the specifications relevant to each assignment. In a discussion thread about a problem with a particular task, you can link to a document that provides a detailed description of how to perform that task.

The Links service provides a means for the application developer to set up source objects (for example, the Discussions service) and target objects (for example, a document), thus enabling your users to create links between the two objects.

There are three actions associated with the Links service: create, delete, and manage. The manage action includes the create and delete actions.

The following custom JSF components are included in the service:

- **Links Detail Button**: This displays an icon and (optionally) a hyperlink that users click to open the Links Panel. To use the Links Detail Button, you must also include the Links - Dialog task flow as a region on the page.
- **Links Detail Menu Item**: This adds a menu item that opens the Links Panel. You can embed this item in an ADF menu. To use the Links Detail Menu Item, you must also include the Links - Dialog task flow as a region on the page.
- **Links Status Icon**: This has two versions:

  The gray **Links** icon (Figure 43–1) indicates that no links are present in the Links dialog.

Figure 43–1 The Links Icon (No Links Present)



  The gold **Links** icon (Figure 43–2) indicates that links are present in the Links dialog.

Figure 43–2 The Links Icon (Links Present)



You can link **from** the following objects:

- Announcements
- Discussions
- Documents
- Events
- Lists
- Pages
- Any object that has a custom JSF component (such as the Links Detail button) bound to it

You can link **to** the following **new** objects:

- Discussions
- Documents
- Events
- Notes
- URLs

You can link **to** the following **existing** objects:

- Announcements
- Discussions
- Documents
- Events

Note: The Lists and Notes services are available in WebCenter Portal: Spaces only.

Links are *not* available for the other WebCenter Portal services, such as Mail and People Connections.

The Links service supports bidirectional links between objects. For example, when you create a link from a discussion topic to a document, a link from the document back to the discussion topic also is created. Similarly, when you *delete* the link from the discussion topic to a document, the link from the document back to the discussion topic is automatically deleted. Bidirectional linking is not available for URLs, notes, and specific list rows.

43.1.2 Requirements for the Links Service

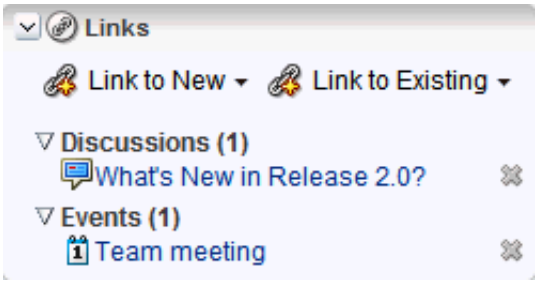
Links automatically recognize any WebCenter Portal service in your application. After you have configured a service in your application, you can add links. However, links work only on secured pages. Links icons do not appear on unsecured pages. For more information, see Section 43.2.3, "Setting Security for the Links Service."

43.1.3 What Happens at Runtime

Linking provides an easy way for you to share information with your social network. Linking can help you realize a significant reduction of wasted time and effort normally spent looking for information.

The Links service exposes its features through a Links dialog (Figure 43–3), accessible wherever the **Links** icon (Figure 43–1) appears in your application.

Figure 43–3 Links Dialog



With the Links service, you can do the following:

- Link an object (such as a page) to an existing object (such as a discussion topic) by clicking the **Link** icon, selecting **Link to Existing**, choosing the resource **Discussions**. Optionally, you can choose a specific forum and click the topic title to choose a link.
- Link an object (such as a discussion topic) to a new object (such as a new note or URL) by clicking the **Link** icon for discussion topic, selecting **Link to New**, and choosing either **Note** or **URL**.
- Create multiple links from one object.
- Delete a link.

For more information about the Links service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

43.2 Basic Configuration for the Links Service

This section describes the steps required for adding the Links service to your application. It includes the following subsections:

- Section 43.2.1, "Setting up Connections for the Links Service"
- Section 43.2.2, "Adding the Links Service at Design Time"
- Section 43.2.3, "Setting Security for the Links Service"
- Section 43.2.4, "Troubleshooting the Links Service"

43.2.1 Setting up Connections for the Links Service

The Links service requires a connection to the database where the WebCenter Portal schema is installed. The link map (that is, relationship information such as what object is linked to what other object) is stored in the database.

For details about setting up a database connection, see Section 7.2.2, "Setting Up a Database Connection."

43.2.2 Adding the Links Service at Design Time

This section explains a basic incorporation of the Links service. It includes the following subsections:

- Section 43.2.2.1, "Links Service Task Flows."
- Section 43.2.2.2, "How to Add the Links Service to your Application."

43.2.2.1 Links Service Task Flows

The Links service includes one task flow: Links Dialog.

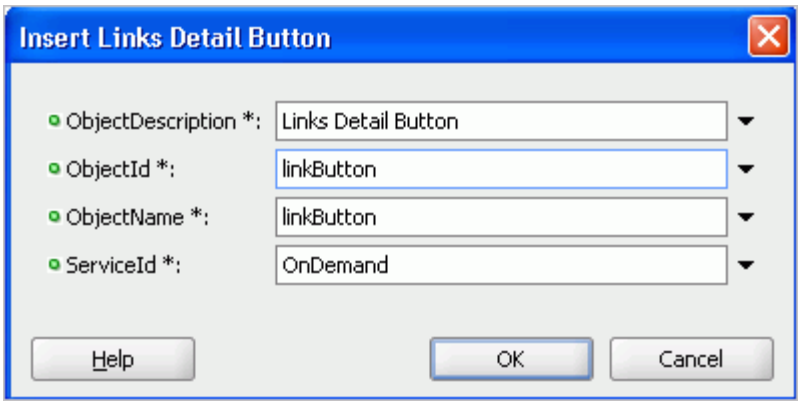
43.2.2.2 How to Add the Links Service to your Application

To add the Links Dialog task flow to your Framework application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application.
2. Ensure that you have set up the database connection to a database with the WebCenter Portal schema installed (Section 43.2.1).
3. Open the page on which you want to add the Links service.
4. In the Component Palette, click **WebCenter Portal - Links Service**.
5. Drag and drop the **Links Detail Button** component onto your page inside the `panelGroupLayout`.

   The button is placed inside of a `panelGroupLayout` for the purposes of this example only. It is not required that you always place the button inside a `panelGroupLayout`.
6. In the Insert Links Detail Button dialog (Figure 43–4), enter a unique object description, ID, and name.

Figure 43–4 Insert Links Detail Button Wizard



The properties in this dialog include:

- **objectDescription**: The description of the object to which you are binding the Links Detail Button
- **objectId**: A unique ID that identifies the object to which you are binding the Links Detail Button
- **objectName**: The name of the object to which you are binding the Links Detail Button
- **serviceId**: An application-wide ID that identifies your application

7. In the **ServiceId** field, enter `OnDemand`.

> **Note:** The Links service combines the `serviceId` and `objectId` to uniquely identify the object to which you bind the Links Detail Button.

8. Click **OK**.

   The new button displays in your page source (Figure 43–5).

Figure 43–5 The Link Detail Button in Your Page Source

```
<af:form><af:panelGroupLayout id="top" layout="horizontal" halign="end"
inlineStyle="background-color:cyan;width:100%">
    <rel:linksDetailButton objectDescription="Link Details Button"
                           serviceId="OnDemand" objectId="linkButton"
                           objectName="linkButton"/>
</af:panelGroupLayout>
```

9. In the Resource Palette, open **My Catalogs**, then open the **Task Flows** folder.
10. Drag and drop the **Links Dialog** task flow next to the Link Detail button on your page, and select **Region** from the context menu.
11. Save and run your page to the browser.

43.2.3 Setting Security for the Links Service

The Links service requires secured pages. Links icons do not appear on unsecured pages.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

43.2.4 Troubleshooting the Links Service

This section describes common problems and solutions for the Links service.

Problem

The **Links** icon does not appear.

Solution

The Links service requires a database connection to the WebCenter Portal schema, where links information is stored. Make sure that you have created the connection to the database and made it available in the Application Resources panel of the Application Navigator. If the connection is available in the Resource Palette but not in Application Resources, then simply drag the connection from the Resource Palette to the Connections folder in Application Resources.

Problem

Existing links appear, but you are not able to create new links or delete existing links.

Solution

The `RelationshipPermission` permission is automatically granted to authorized users when a service is consumed. Verify that this permission has been granted. For information, see Section 69.3.2, "Automated Security Grants for WebCenter Portal Services."

43.3 Advanced Information for the Links Service

This section describes optional features available with this service. It includes the following subsection:

- Section 43.3.1, "Using the Links Service REST APIs"

43.3.1 Using the Links Service REST APIs

WebCenter Portal provides REST APIs to support the Links service. Use the Links service REST APIs to post links between two objects. For example, you could create a link between an event and a document in a space.

This section describes the REST APIs associated with the Links service. It includes the following subsections:

- Section 43.3.1.1, "Links Entry Point"
- Section 43.3.1.2, "Links Resource Type Taxonomy"
- Section 43.3.1.3, "Security Considerations"
- Section 43.3.1.4, "Links Resource Types"

For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

43.3.1.1 Links Entry Point

Each REST service has a link element within the resource index that provides the entry point for that service. To find the entry point for the Links service, find the link element with a `resourceType` of:

```
urn:oracle:webcenter:links
```

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Links service.

For more information about the resource index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

43.3.1.2 Links Resource Type Taxonomy

When the client has identified the entry point, it then can navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 54.5.2.1, "Resource Type."

The taxonomy for the Links service is:

```
urn:oracle:webcenter:links
   urn:oracle:webcenter:links:results
```

Beyond the service entry points, URL templates allow clients to pass query parameters to customize their requests and control the form of returned data.

43.3.1.3 Security Considerations

Authentication is required before using Links REST API methods.

For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

43.3.1.4 Links Resource Types

This section provides information about each resource type. It includes the following subsection:

- Section 43.3.1.4.1, "urn:oracle:webcenter:links:results"

43.3.1.4.1 urn:oracle:webcenter:links:results Use this resource type to identify the URI to use to create (`POST`) a link between two objects

The request is represented by the URL, and the response is a link, each with metadata to help the end user choose an item to drill down and cross links to the owning REST services, if available. If the owning REST service is unavailable, then an HREF link is provided.

Navigation Paths to results

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   links
```

```
      results

resourceindex
   spaces
      spaces:resourceindex
         spaces:links
            results
```

Supported Methods for results

The following method is supported by this resource:

- `POST`
  - **request - body:** a source and a target object, each of which has a `serviceId`, `resourceId`, and `resourceName`; if the target is a URL link, then the target object also contains a `resourceUrl`, **Parameters**: `serviceId`, `resourceId`, `resourceName`
  - **response - body:** Status of "204 No Content" if successful, or "403 Forbidden" if the user has no permission to create a link

where:

- `serviceId={serviceId}`

  Optional: The service ID of the item, such as `services.oracle.webcenter.collab.calendar.community`

- `resourceId={resourceId}`

  The resource ID, such as `s833cddc4_caa5_416c_87e9_d702ef870b43;;3b0c6edb`

- `resourceName={resourceName}`

  The resource name, such as `New Event`.

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to From results

Table 43–1 lists the resource types that the client can link to from this resource.

Table 43–1 Related Resource Types for links

rel	resourceType
self	urn:oracle:webcenter:links:results

44

Integrating the Tags Service

This chapter explains how to integrate the Tags service in a WebCenter Portal: Framework application at design time.

For more information about managing and including tags, see:

- "Setting up the WebCenter Portal Repository" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Tags Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

> **See Also:** You must register a resource viewer to enable custom resources to be rendered using Search or Tags, or to make the resources linkable to and from each other. For more information, see Section 7.3, "Extending Your Framework Application with Custom Components."

This chapter includes the following sections:

- Section 44.1, "Introduction to the Tags Service"
- Section 44.2, "Basic Configuration for the Tags Service"
- Section 44.3, "Advanced Information for the Tags Service"

44.1 Introduction to the Tags Service

Tags enable users to apply their own meaningful terms to items, making those items more easily discoverable in search results and the Tag Center. The Tag Center is a page that displays the interactions among all the tags, tagged items, and their taggers in an application.

Having multiple users tag objects contributes to the collective knowledge, which makes searching much more relevant. WebCenter Portal search takes advantage of the knowledge captured by tagging by indicating the relevance of results based on the quality and frequency of their applied tags.

Tags let the *users* of data (instead of the *publishers* of data) classify information. In this way, tags act as a personal productivity tool and a method to increase searchability for everyone.

For example, suppose your application includes a component that provides a view of departmental human resources contacts. If you enable tags for this component, then a user who comes to it can bookmark it for themselves and assign tags like HR or contacts or department to help others find it as well.

The Tags service is available for pages, documents, and custom objects. Creating a tag automatically publishes the event to the Activity Stream.

44.1.1 Understanding the Tags Service

You can apply one or more meaningful terms, called *tags*, to remind yourself and alert other users of the content they might expect to find at the tagged location. Anywhere you see the **Tags** icon (Figure 44–1), you can apply a tag.

Figure 44–1 The Tags Icon



Tags let you apply your own classifications to items. For example, you could apply the tag `phone` to a product page that provides useful information about new phones. When you or other users search for phone, the tagged page displays in the search results.

When you access the Tag Center, you see all users who applied the same tag anywhere else in the application. Within the Tag Center, a tag cloud (Figure 44–2) displays all currently applied tags.

Figure 44–2 A Tag Cloud



A tag cloud is a visual depiction of all tags. Tags are presented according to the frequency of their use—the larger the font, the more the tag has been applied to items. Click a tag in the tag cloud to run a search that returns a list of all items that use the tag.

44.1.2 Requirements for the Tags Service

The Tags service requires that you set up the WebCenter Portal schema and create a database connection to the schema. Tag information is stored in the database. For details, see Section 7.2.2, "Setting Up a Database Connection."

> **Note:** Typically, when you add tags in a Framework application, you also should add a search toolbar to enable searching for the objects with the tags. To add the search toolbar, follow the instructions in Section 45.2.3, "Adding the Search Service at Design Time."

The Tags service requires security. If users are not authenticated, then they cannot tag objects. If the application is not secured, then pages are not returned in search results for tagged items.

To enable the ability to tag *documents*, you must have the Document Manager task flow in the Framework application. For more information, see Chapter 30, "Integrating the Documents Service."

44.1.3 What Happens at Runtime

At runtime, when you search for a tag word, results are returned in the Tagged Items section of the search results. By default, tagged items that are accessed through their tags are rendered in a full screen replacing the same window.You can override this and have it open in a popup window.

You can tag custom components, find those tags either through a search or in the Tag Center, and access tagged items through their associated tags.

For services exposing resources to be tagged (and therefore searched and viewed) WebCenter Portal provides a Resource Action Handling framework. The Tags service uses this framework to allow acting on a search result. For more information, see Section 7.3, "Extending Your Framework Application with Custom Components."

For more information about the service at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

44.2 Basic Configuration for the Tags Service

This section describes required steps for adding this service to your application. It includes the following subsections:

- Section 44.2.1, "Setting up Connections for the Tags Service"
- Section 44.2.2, "Adding the Tags Service at Design Time"
- Section 44.2.3, "Setting Security for the Tags Service"

44.2.1 Setting up Connections for the Tags Service

The Tags service requires a connection to the database with the WebCenter Portal schema installed.

For details about setting up a database connection, see Section 7.2.2, "Setting Up a Database Connection."

44.2.2 Adding the Tags Service at Design Time

To enable basic tagging, you must add either the **Tagging Button** component or the **Tagging Menu Item** component and the Tagging - Dialog task flow. In addition, to tag *custom* objects, you must register a resource viewer to visualize the tagged component when it is discovered by users.

This section describes how to enable tagging by adding the **Tagging Button** and the Tagging - Dialog task flow. It includes the following subsections:

- Section 44.2.2.1, "Tags Service Components"
- Section 44.2.2.2, "Tags Service Task Flows"
- Section 44.2.2.3, "How to Add the Tags Service to your Page"
- Section 44.2.2.4, "How to Modify the Tags Service Task Flow Parameters"

44.2.2.1 Tags Service Components

The following JSF components are included in the service:

- **Tagging Button**: When you add a Tagging Button, a Tags link on the page enables you to invoke the Tag This Item (or Tag This Page) dialog at runtime.
- **Tagging Menu Item**: When you add a Tagging Menu Item, a menu option on the page enables you to invoke the Tag This Item (or Tag This Page) dialog at runtime.

You can configure attributes in the Tags components. For example, the `SharedParam` attribute lets you disable the **Shared** checkbox. (With the `default` value, the checkbox is enabled, so users can check it or uncheck it at will. With the `always` value, the checkbox is disabled in the true state, so all tags are public. With the `never` value, the checkbox is disabled in the false state, so all tags are private.) As another example, the `ResourceScope` attribute lets users tag objects in a different scope, such as a list of search results with the Tagging button next to each result, where the search is done in the default scope but the results could be in other scopes. For more information about the component attributes, see the online help.

44.2.2.2 Tags Service Task Flows

Table 44–1 lists the Tags service task flows.

Table 44–1 Tags Service Task Flows

Task Flow	Definition
Tagging - Dialog	This task flow displays a dialog that appears to tag a particular object. This does not have any visible user interface rendering when dropped onto a page. It only becomes visible when the tagging button is clicked.
Tagging - Personal View	This task flow shows the tags created by the current user, objects tagged by those tags, and a cloud view of the tags. A service ID input parameter enables you to view tagged items of interest. Because this is a non-public task flow, it requires authentication.
Tagging - Related Links	For an object identified by a unique ID, this task flow lists all the other objects that are tagged with similar tags. The other objects listed include those that are similarly tagged; that is, their tag "sets" have an intersection of at least one. The list is ordered by the number of tags in the intersection.
Tagging - Tag Cloud	This task flow displays a tag cloud, which is a visual depiction of all the tags used. Tags are presented according to the frequency of their use. More frequently used tags display in bold fonts and varying font sizes—the larger the font, the more the tag has been applied. When you click a tag in the tag cloud, you are redirected to Tag Center, where that tag is selected. This task flow appears on a page with tagged items. It provides Resource Action Handling (RAH) links for all tags in the page.
Tagging - Tag Selection	Similar to the Tag Cloud task flow, the Tag Selection task flow also displays a tag cloud. However, when you select a tag, it marks the tag as selected, and you see the results in the Related Resources task flow. You are not redirected to the Tag Center. With this task flow and the Tagging - Tagged Items task flow, you can click a tag in the tag cloud to see the items that use the tag. (In other words, selecting a tag in the Tagging - Tag Cloud task flow sends the tag word as an event to all interested event consumers, such as the Tagging - Tagged Items task flow.)

Table 44–1 (Cont.) Tags Service Task Flows

Task Flow	Definition
Tagging - Tagged Items	This task flow displays items pages, or documents that have been tagged. It can refine itself to show only the items tagged with the tag passed to it. This task flow can listen for events sent by the Tagging - Tag Cloud task flow.

See Also: Section 44.2.2.4, "How to Modify the Tags Service Task Flow Parameters"

44.2.2.3 How to Add the Tags Service to your Page

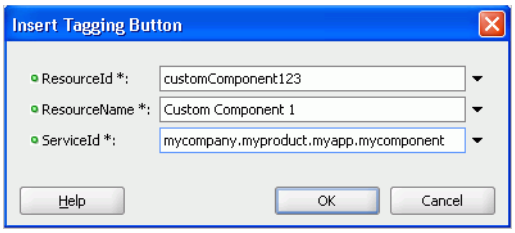
To add the Tags service to your WebCenter Portal: Framework application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application.
2. Ensure that you have set up a database connection to a database with the WebCenter Portal schema installed.
3. Open the page where you included the search toolbar.
4. On the Component Palette (All Pages), select the **Tagging Button** and drop it on the page, inside a `PanelGroupLayout`. Make it the first child of the `PanelGroupLayout` by placing it to the left of the search toolbar. The Tagging Button enables users to start tagging.
5. Fill out the Insert Tagging Button dialog. Table 44–2 provides descriptions and example values for the fields in the dialog.

Table 44–2 Fields in the Insert Tagging Button Dialog

Field	Description
`ResourceId`	The ID that uniquely identifies the object to which you are binding the Tagging Button. The value in this field need not be static. For example, you could use EL to insert a unique value for each row in a table. For example: ▪ To tag pages, enter `#{facesContext.viewRoot.viewId}` ▪ To tag custom components, enter `customComponent123`
`ResourceName`	The name of the object to which you are binding the Tagging Button. The value in this field need not be static. For example, you could use EL to insert a unique value for each row in a table. ▪ To tag pages, enter `#{facesContext.viewRoot.viewId}` ▪ To tag custom components, enter `Custom Component 1`
`ServiceId`	An application-wide ID. (With custom components, you add this to `service-definition.xml` when adding the resource viewer.) ▪ To tag pages, enter `oracle.webcenter.page` ▪ To tag custom components, enter `mycompany.myproduct.myapp.mycomponent`

Figure 44–3 Insert Tagging Button Dialog for a Custom Component



6. Click **OK**.

> **Note:** The Tagging Button includes additional properties you can set. For example, `SharedParam` lets you disable the **Shared** checkbox. (With the `default` value, the checkbox is enabled, so users can check it or uncheck it at will. With the `always` value, the checkbox is disabled in the true state, so all tags are public. With the `never` value, the checkbox is disabled in the false state, so all tags are private.)

7. From the Resource Palette, expand **WebCenter Portal - Services Catalog** and **Task Flows**.
8. Drag and drop **Tagging Dialog** from the Resource Palette on the page, optionally adding a redirection URL for anonymous page login.
9. When prompted, select **Region** as the way to create the task flow.
10. For custom resources, follow the steps described in Section 7.3, "Extending Your Framework Application with Custom Components" to register a resource viewer. For services exposing resources to be tagged (and therefore searched and viewed) WebCenter Portal provides a Resource Action Handling framework. The Tags service uses this framework to allow acting on a search result.

In the Source view where you added the Tags service, you should see something similar to Example 44–1.

Example 44–1 Tagging Button and Tagging - Dialog Task Flow Source

```
<af:form id="f1">
        <af:region value="#{bindings.searchtoolbar1.regionModel}" id="r1"/>
        <tag:taggingButton resourceId="#{facesContext.viewRoot.viewId}"
                           serviceId="oracle.webcenter.page"
                           resourceName="#{facesContext.viewRoot.viewId}"
                           id="tb1"/>
        <af:region value="#{bindings.tagginglaunchdialog1.regionModel}"
                   id="r2"/>
      </af:form>
```

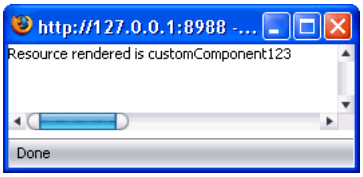
> **Note:** Even if there are multiple Tagging Buttons on the page, only one Tagging Dialog `af:region` is needed.

Run the page with the **Tagging Button** and Tagging - Dialog task flow. When the page renders in the browser, click the **Tags** link.

1. When the dialog appears, enter the tag `home` and click **Save**. Note that every user of the system can perform this same operation and assign the same tag, which means that you get a weighted collection of tags.

2. In the search toolbar, type the search term `home`, and click the **Search** icon.
3. You should see `home` as a search result under Tags and `Custom Component 1` as a search result under Tagged Items.
4. Click `Custom Component 1` and a dialog appears with the resource viewer you just created (Figure 44–4).

Figure 44–4 Resource Viewer



44.2.2.4 How to Modify the Tags Service Task Flow Parameters

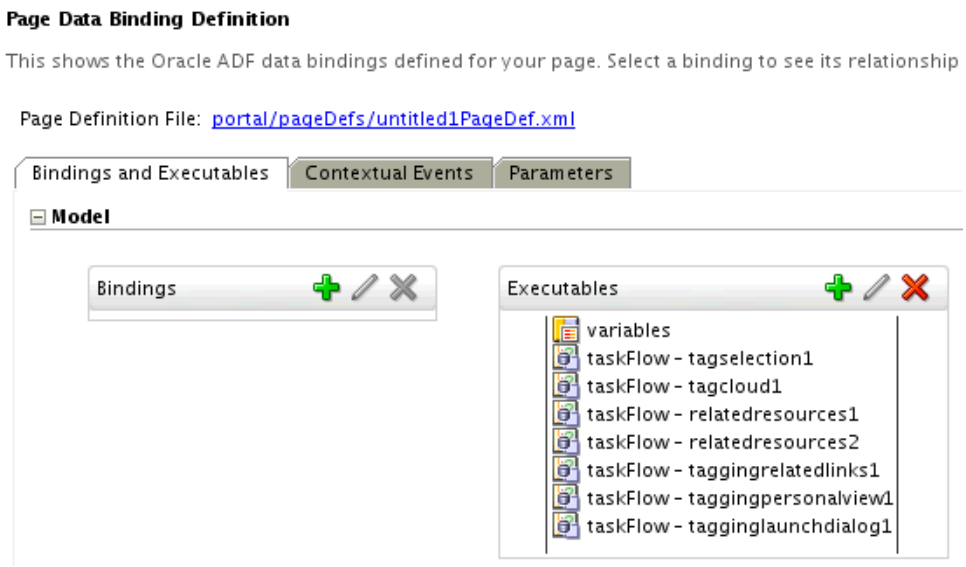
Some Tags service task flows have optional task flow binding parameters.

> **Note:** You should not change the Tags service task flow properties unless you want to show items from a different resource or different service.

To see and adjust task flow binding parameter values after you have placed a task flow on a page:

1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 44–5 shows an example of Tags task flows in the Executables section.

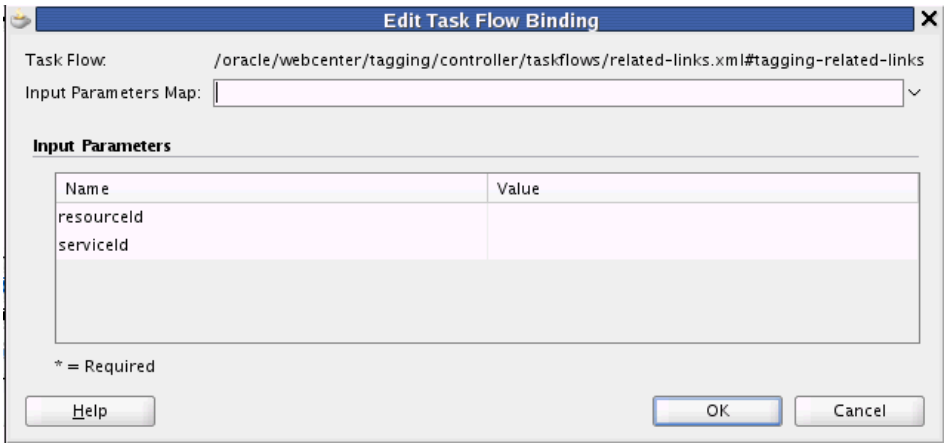
Figure 44–5 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.

4. In the Edit Task Flow Binding dialog Figure 44–6, revise the binding parameter values as required.

Figure 44–6 Edit Task Flow Binding Dialog for Tags - Related Links Task Flow



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

Table 44–3 describes the properties that are unique to the Tags service task flows.

Table 44–3 Tags Service Task Flow Parameters

Property	Description	Task Flow
`resourceId`	Unique ID of the item or resource within a given service that is used to find similarly tagged items. This value is set automatically. Do not change this value unless you want to show items similar to a different resource.	Tagging - Related Links
`serviceId`	This parameter has a different meaning for the Related Links task flow and the Personal View task flow. ▪ In Related Links, it is a `resourceId/serviceId` pair that defines the item that has related links. This value is set automatically. Do not change this value unless you want to show items similar to a different resource belonging to a different service. ▪ In Personal View, it limits the tags to this service; for example, `oracle.webcenter.page`.	Tagging - Related Links Tagging - Personal View
`resourceScope`	Scope of tags. This value is set automatically (either `#{spaceContext.currentSpaceGUID}` or something like `s8bba98ff_4cbb_40b8_beee_296c916a23ed`). Do not change this value.	Tagging - Tag Cloud Tagging - Tag Selection Tagging - Tagged Items
`redirectURL`	Optional redirection URL for anonymous page login. This must include the `/faces` prefix; for example, `/faces#{facesContext.viewRoot.viewId}` If null or empty, then no login is displayed.	Tagging - Dialog

44.2.3 Setting Security for the Tags Service

The Tags service requires security. If users are not authenticated, then they cannot tag objects. If the application is not secured, then pages are not returned in search results for tagged items.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

> **Note:** You must register a resource viewer to allow custom objects to be found using Search or Tags. For information about how the resource viewer's `authorizerClass` determines whether users can view the resource, see Section 7.3, "Extending Your Framework Application with Custom Components."

44.3 Advanced Information for the Tags Service

This section describes optional features available with this service. It includes the following subsections:

- Section 44.3.1, "Using the Tags Service Java APIs"
- Section 44.3.2, "Using the Tags Service REST APIs"
- Section 44.3.3, "Optional Way to Show Tags on Pages"
- Section 44.3.4, "Using the Resource Action Handling Framework to Tag Custom Objects"
- Section 44.3.5, "Troubleshooting the Tags Service"

> **Note:** You can turn up logging levels for the tagging component (as you can with other components) in the `logging.xml` file, found under the WLS home.

44.3.1 Using the Tags Service Java APIs

Custom components can implement the Tags service APIs to create a tagging back end compatible with the WebCenter Portal REST APIs or Tags service task flows and components.

For more information on the Java APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

44.3.2 Using the Tags Service REST APIs

WebCenter Portal provides REST APIs to support the Tags service. Use the Tags service REST APIs to do the following:

- Find all tags or find a subset of tags by criteria
- Get a specific tag
- Delete tags
- Update or rename tags
- Get tagged items
- Untag an item
- Add a new tagged item
- Change tags for a tagged item
- Find all tagged items or find tagged items by a filter, such as a date or a tag word

- Get related user by tag

This section describes the REST APIs associated with the Tags service. It includes the following subsections:

- Section 44.3.2.1, "Tags Entry Point"
- Section 44.3.2.2, "Tags Resource Type Taxonomy"
- Section 44.3.2.3, "Security Considerations"
- Section 44.3.2.4, "Tags Resource Types"

For an introduction to the REST APIs, see Section 54, "Using Oracle WebCenter Portal REST APIs."

44.3.2.1 Tags Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. To find the entry point for the Tags service, find the link element with a `resourceType` of one of the following:

- `urn:oracle:webcenter:tagging:tags`
- `urn:oracle:webcenter:tagging:taggeditems`
- `urn:oracle:webcenter:tagging:users`

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Tags service.

For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

44.3.2.2 Tags Resource Type Taxonomy

When the client has identified the entry point, it then can navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 54.5.2.1, "Resource Type."

The taxonomy for the Tags service is:

```
urn:oracle:webcenter:tagging:tags
   urn:oracle:webcenter:tagging:tag
urn:oracle:webcenter:tagging:taggedItems
   urn:oracle:webcenter:tagging:taggedItem
urn:oracle:webcenter:tagging:users
```

Under users, you can add the resource type for that object, such as peopleprofile.

Beyond the service entry points, URL templates allow clients to pass query parameters to customize their requests and control the form of returned data.

Collection resources in the tags resources support pagination (`itemsPerPage` and `startIndex` for tags and users and `itemsPerPage` for taggedItems). Other query parameters (`search` and `projection`) are not supported.

44.3.2.3 Security Considerations

Authentication is required before using Tags REST API methods.

For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

44.3.2.4 Tags Resource Types

This section provides information about each resource type. It includes the following subsections:

- Section 44.3.2.4.1, "urn:oracle:webcenter:tagging:tags"
- Section 44.3.2.4.2, "urn:oracle:webcenter:tagging:tag"
- Section 44.3.2.4.3, "urn:oracle:webcenter:tagging:taggedItems"
- Section 44.3.2.4.4, "urn:oracle:webcenter:tagging:taggedItem"
- Section 44.3.2.4.5, "urn:oracle:webcenter:tagging:users"

44.3.2.4.1 urn:oracle:webcenter:tagging:tags Use this resource type to identify the URI to use to read (`GET`) tags. The response from a `GET` operation includes all tags, a specific tag, a subset of tags, or a tag related to a specific list of tags or users.

Navigation Paths to tags

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   tags

resourceindex
   spaces
      spaces:resourceindex
         spaces:tags
```

Supported Methods for tags

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters**: `keyword`, `serviceId`, `relatedId`, `user`, `startIndex`, `itemsPerPage`, `shared`, `tags`, `users`
  - **response - body:** tag

where:

- `keyword={keyword for substring match}`

  The keyword for substring matches.

- `serviceId={serviceId}`

  The service ID for which you want to find tags. (null for all)

- `relatedId= {related Tag Id}`
- `user={...}`

  The user ID to find tags for. If null, this uses the logged-in user.

- `shared = {true/false}`

  Returns personal or system tags.

- `tags={TAG}+{TAG}+...+{TAG}`

  Filter by these tags separated by URL-encoded spaces ('+').

- `users = {USER}+{USER}+...+{USER)`

Filter by these users separated by URL-encoded spaces ('+').

For more information, see Section 54.5.2.5, "Templates."

Table 44–4 shows the available parameter combinations for each operation.

Table 44–4 Tags GET Parameters

API Method	tags	tagWords	UserId	UserFilter	ServiceId	shared	scope	keyword
findRelatedSystemTags	X	X		X			X	
findPopularTagsCommon			X		X	X		X
findPopularTags (with GUID)			X			X	X	X
findPopularTags			X			X		X

Resource Types Linked to From tags

Table 44–5 lists the resource types that the client can link to from this resource.

Table 44–5 Related Resource Types for tags

rel	resourceType
self	urn:oracle:webcenter:tagging:tags
Previous/Next	urn:oracle:webcenter:tagging:tags

44.3.2.4.2 urn:oracle:webcenter:tagging:tag Use this resource type to identify the URI to use to read (`GET`), rename (`PUT`), and delete (`DELETE`) a tag. The response from a `GET` operation includes all tags, a specific tag, a subset of tags, or a tag related to a specific list of tags or users.

Navigation Paths to tag

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   tags
      tag

resourceindex
   spaces
      spaces:resourceindex
         spaces:tags
            spaces:tag
```

Supported Methods for tag

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters**: none
  - **response - body:** tag
- `PUT`
  - **request - body:** tag
  - **response - body:** tag
- `DELETE`

- **request - body:** none
- **response - body:** none

```
<tag>
<name>newId</name>
</tag>
```

Resource Types Linked to From tag

Table 44–5 lists the resource types that the client can link to from this resource.

Table 44–6 Related Resource Types for tag

rel	resourceType
self	urn:oracle:webcenter:tagging:tag
	urn:oracle:webcenter:tagging:taggedItems
	urn:oracle:webcenter:tagging:users
	urn:oracle:webcenter:tagging:tags

44.3.2.4.3 urn:oracle:webcenter:tagging:taggedItems Use this resource type to identify the URI to use to read tagged items and their related resources (`GET`) or to add tagged items (`POST`).

The response from a `GET` operation includes a tagged item or all tagged items, which can be filtered by date, tag words, and so on. Each tag includes links used to operate on that tag. The response from a `POST` operation includes the tagged items that were created in this collection of tags and a link to operate on them.

Navigation Paths to taggedItems

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   taggedItems

resourceindex
   spaces
      spaces:resourceindex
         spaces:taggedItems
```

Supported Methods for taggedtems

The following methods are supported by this resource:

- `POST`
  - **request - body:** taggedItem
  - **response - body:** taggedItem

```
<taggedItem>
<serviceId>oracle.webcenter.page</serviceId>
<resourceId>/oracle/webcenter/page/somepath/someotherpath/a.jspx</resourceId>
<tags>foo bar boo far</tags>
<resourceTitle>This is renamed by me to be foofoo</resourceTitle>
<scopeGuid>(optional) scope for the item</scopeGuid>
<shared>(optional) make the item private </shared>
</taggedItem>
```

- `GET`
  - **request - body:** none, **Parameters**: `serviceId`, `keyword`, `tags`, `users`, `dt`, `shared`, `itemsPerPage`
  - **response - body:** tagged items

where:

- `serviceId={serviceId}`

  The service ID of the item.

- `keyword={keyword}`

  A keyword in the resource title.

- `tags={TAG}+{TAG}+...+{TAG}`

  Filter by these tags, separated by URL-encoded spaces ('+').

- `users={USER}+{USER}+...+{USER}`

  Filter by these users, separated by URL-encoded spaces ('+').

- `dt={DD-MM-CCYY}`

  Filter bookmarks by this date, which defaults to the most recent date on which bookmarks were saved.

- `shared = {true/false}`

  System or personal resources only.

For more information, see Section 54.5.2.5, "Templates."

Table 44–7 shows the available parameter combinations for each operation.

Table 44–7 TaggedItems Parameters

API Method	serviceId	resourceId	date	shared	tagWords	userFilter	keyword	scope
findRelatedSystemResources	X	X						
findUpdatedResources			X					X
findSystemResources				X	X	X		X
findFilterPersonalResources	X						X	X
findPersonalResources				X	X	X		X
findFilteredPersonalResources (no keyword)	X							X

Resource Types Linked to From taggedItems

Table 44–8 lists the resource types that the client can link to from this resource.

Table 44–8 Related Resource Types for taggedItems

rel	resourceType
self	urn:oracle:webcenter:tagging:taggedItems
Previous/Next	urn:oracle:webcenter:tagging:taggedItems

44.3.2.4.4 urn:oracle:webcenter:tagging:taggedItem Use this resource type to identify the URI to use to read tagged items and their related resources (`GET`), add tagged items

(POST), update tags for a tagged item (PUT), and remove all tags from an item (DELETE).

The response from a GET operation includes a tagged item or all tagged items, which can be filtered by date, tag words, and so on. Each tag includes links used to operate on that tag. The response from a POST operation includes the tagged item that was created in this collection of tags and a link to operate on that tagged item.

Navigation Paths to taggedItem

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   taggedItem

resourceindex
   spaces
      spaces:resourceindex
         spaces:taggedItem
```

Supported Methods for taggedItem

The following methods are supported by this resource:

- GET
  - **request - body:** none, **Parameters**: serviceId, keyword, tags, users, dt, shared, itemsPerPage
  - **response - body:** tagged item
- PUT
  - **request - body:** taggedItem
  - **response - body:** taggedItem

```
<taggedItem>
<tags>foo bar boo far</tagWords>
</taggedItem>
```

- DELETE
  - **request - body:** none
  - **response - body:** none

Resource Types Linked to From taggedItem

Table 44–8 lists the resource types that the client can link to from this resource.

Table 44–9 Related Resource Types for taggedItem

rel	resourceType
self	urn:oracle:webcenter:tagging:taggedItem
	urn:oracle:webcenter:tagging:taggedItems
alternate	The resource type depends on the type of resource tagged; for example, a page would be urn:oracle:webcenter:page:page

44.3.2.4.5 urn:oracle:webcenter:tagging:users Use this resource type to get a list of people related to a given list of tags.

Navigation Paths to users

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   taggingusers

resourceindex
   spaces
      spaces:resourceindex
         spaces:taggingusers
```

Supported Methods for users

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters**: `startIndex`, `itemsPerPage`, `tags`, `users`
  - **response - body:** people profile

where:

- `tags={TAG}+{TAG}+...+{TAG}`

  Filter by these tags separated by URL-encoded spaces ('+').

- `users = {USER}+{USER}+...+{USER)`

  Filter by these users separated by URL-encoded spaces ('+').

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to From users

Table 44–5 lists the resource types that individual users can link to from this resource.

Table 44–10 Related Resource Types for users

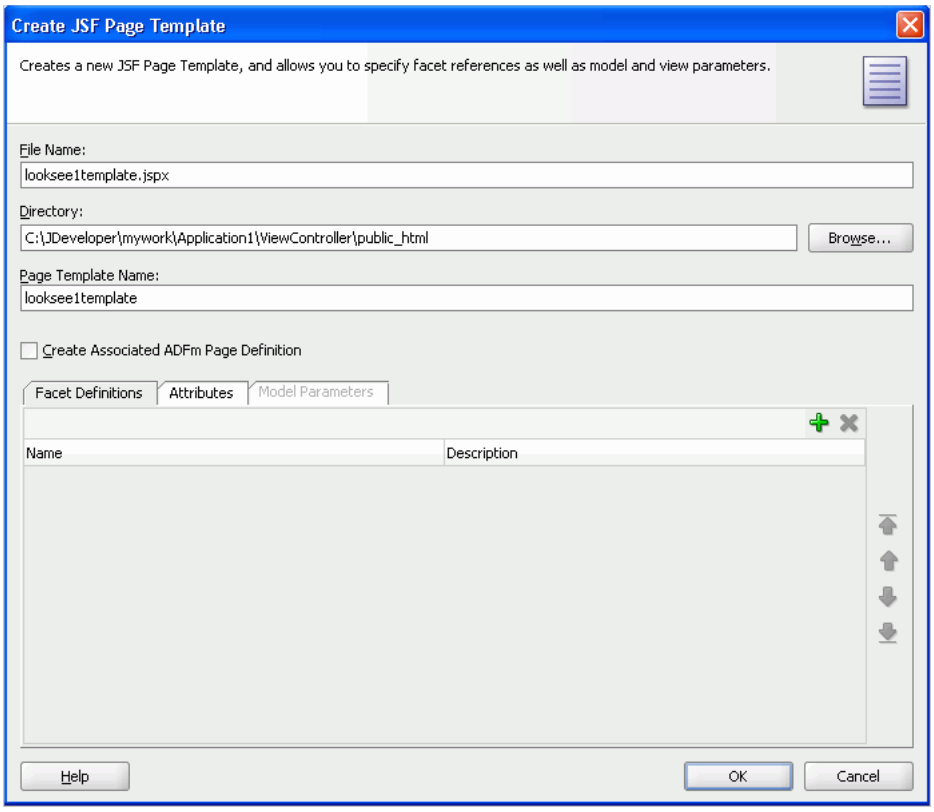
rel	resourceType
self	urn:oracle:webcenter:tagging:users
Previous/Next	urn:oracle:webcenter:tagging:users

44.3.3 Optional Way to Show Tags on Pages

While it is possible to add the Tagging Button and task flows on a page-by-page basis, typically you want to add the search toolbar and tags as part of a page template that you can apply whenever building a page that requires it. To add tags through a page template:

1. Open the project where you plan to create pages with the Tags service.
2. Right-click the project name and select **New** from the context menu.
3. Under **Categories**, select **JSF** and, under **Items**, select **JSF Page Template**.
4. For **File Name**, enter `looksee1template`.

Figure 44–7 Create JSF Page Template Dialog



5. Click **OK**.
6. Go to the Source view and modify the code to be similar to that in Example 44–2.

Example 44–2 Sample JSF Page Template

```
<af:pageTemplateDef var="attrs">
  <af:xmlContent>
    <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
      <display-name>looksee1template1</display-name>
      <facet>
        <facet-name>content</facet-name>
      </facet>
    </component>
  </af:xmlContent>
  <af:panelGroupLayout layout="vertical">
    <!-- Our toolbar -->
    <af:panelGroupLayout id="toolbar" layout="horizontal" halign="end"
                         inlineStyle="background-color:cyan;width:100%">
    </af:panelGroupLayout>
    <af:panelStretchLayout startWidth="300"
                           inlineStyle="height:600px; width:100%;">
      <f:facet name="start">
        <af:panelSplitter id="sidebar" orientation="vertical">
          <f:facet name="first">
            </f:facet>
          <f:facet name="second">
          </f:facet>
        </af:panelSplitter>
      </f:facet>
```

```
      <f:facet name="center">
        <af:facetRef facetName="content"/>
      </f:facet>
    </af:panelStretchLayout>
  </af:panelGroupLayout>
</af:pageTemplateDef>
```

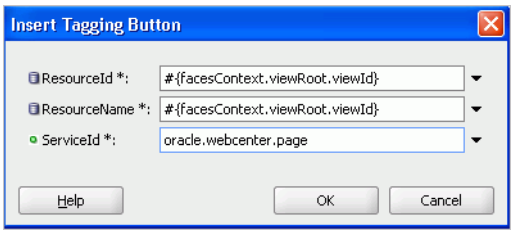
7. On the Component Palette, select the **Tagging Button** and drop it on the page, inside the `PanelGroupLayout`. Make it the first child of the `PanelGroupLayout`.
8. Fill out the Insert Tagging Button dialog as shown in Figure 44–8.

   Table 44–11 provides descriptions and example values for the fields in the dialog.

Table 44–11 Fields in the Insert Tagging Button Dialog for a Page

Field	Description
`ResourceId`	Unique identifier of your page. In this case, because the resource ID is meant for a page, enter `#{facesContext.viewRoot.viewId}`.
`ResourceName`	Name of the page. In this case, enter `#{facesContext.viewRoot.viewId}`. (You can enter a more user-friendly name.)
`ServiceId`	Service identifier that you add to `service-definition.xml` when adding the resource viewer. In this case, enter `oracle.webcenter.page`. Note: Because `oracle.webcenter.page` is available in the Page service library, nothing must be added to `service-definition.xml`.

Figure 44–8 Insert Tagging Button Dialog for a Page



9. Click **OK**.
10. From the Resource Palette, expand the **WebCenter Portal - Services Catalog** and the **Task Flows**.
11. Drag **Tagging Dialog** from the Resource Palette and drop it on the page inside of the `panelGroupLayout`.
12. When prompted, select **Region** as the way to create the task flow.
13. Drag **Search Toolbar** and drop it after the **Tagging Dialog** you just added to the page template in the `panelGroupLayout`.
14. When prompted, select **Region** as the way to create the task flow.
15. Drag **Document Library - Recent Documents** and drop it in the second facet of the `panelSplitter` with identifier `sidebar`.

16. When prompted, select **Region** as the way to create the task flow.
17. In the Edit Task Flow Binding dialog, for `connectionName`, enter the name of the connection to your document library's content repository; for example, `${'MyFileSystem'}`. For information on how to set up a connection to your content repository, see Section 27.2, "Configuring Content Repository Connections."
18. The source for your page template should now look similar to Example 44–3.

Example 44–3 JSF Page Template with Tags and Search

```
<af:pageTemplateDef var="attrs">
  <af:xmlContent>
    <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
      <display-name>looksee1template1</display-name>
      <facet>
       <facet-name>content</facet-name>
      </facet>
    </component>
  </af:xmlContent>
  <af:panelGroupLayout layout="vertical">
   <!-- Our toolbar -->
   <af:panelGroupLayout id="top" layout="horizontal" halign="end"
              inlineStyle="background-color:cyan;width:100%">
    <tag:taggingButton resourceId="#{facesContext.viewRoot.viewId}"
                    resourceName="#{facesContext.viewRoot.viewId}"
                       serviceId="oracle.webcenter.page"/>
     <af:region value="#{bindings.tagginglaunchdialog1.regionModel}"
                id="tagginglaunchdialog1"/>
     <af:region value="#{bindings.searchtoolbar1.regionModel}"
                id="searchtoolbar1"/>
   </af:panelGroupLayout>
    <af:panelStretchLayout startWidth="300"
                          inlineStyle="height:600px; width:100%;">
     <f:facet name="start">
     <af:panelSplitter orientation="vertical">
      <f:facet name="first">
       <af:region value="#{bindings.taskservicetaskflowdefinition1.regionModel}"
                  id="taskservicetaskflowdefinition1"
                  inlineStyle="position:absolute; left:0px; width:100%;
                   height:100%"/>
      </f:facet>
      <f:facet name="second">
       <af:region value="#{bindings.doclibrecentdocuments1.regionModel}"
                  id="doclibrecentdocuments1"/>
      </f:facet>
       </af:panelSplitter>
      </f:facet>
      <f:facet name="center">
       <af:facetRef facetName="content"/>
      </f:facet>
    </af:panelStretchLayout>
   </af:panelGroupLayout>
  </af:pageTemplateDef>
```

19. Create two new pages based upon the page template you just created. In the Create JSF Page dialog, select `looksee1template1` from the **Use Page Template** list.

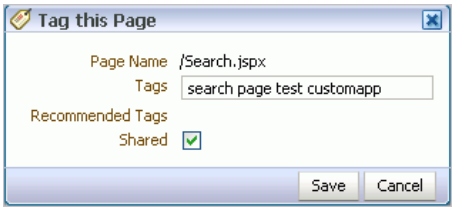
- Create a page called `Search.jspx` and add the **Search** task flow in the center facet.
- Create a page called `Documents.jspx` and add the **Document Library - List View** task flow as the center facet. In the Edit Task Flow Binding dialog, for `connectionName`, enter the name of the connection to your document library's content repository; for example, `${'MyFileSystem'}`. For information on how to set up a connection to your content repository, see Section 27.2, "Configuring Content Repository Connections."

20. Save your pages.

To run your pages with tags:

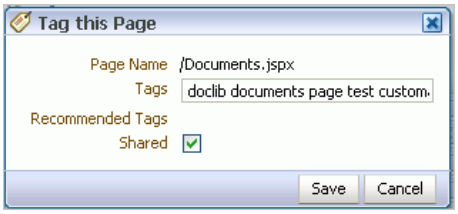
1. In the Applications Navigator, run the `Search.jspx` page.
2. After you log in and the page comes up, click the **Tags** link.
3. In the Tag this Page dialog, enter `search page test customapp` for **Tags** (Figure 44–9).

Figure 44–9 Tag this Page Dialog for Search Page



4. Click **Save**.
5. Navigate to `Documents.jspx` (or run it separately).
6. Click the **Tags** link.
7. In the Tag this Page dialog, enter `doclib documents page test customapp` for **Tags** (Figure 44–10).

Figure 44–10 Tag this Page Dialog for Documents Page



8. Click **Save**.
9. In the search toolbar for the `Documents.jspx` page, enter `search` and click the **Search** icon.
10. Under **Tags**, click `search`.

44.3.4 Using the Resource Action Handling Framework to Tag Custom Objects

For services exposing resources to be tagged (and therefore searched and viewed) WebCenter Portal provides a Resource Action Handling framework. For example, rows in a table can be tagged.

To register a resource viewer, see Section 7.3, "Extending Your Framework Application with Custom Components."

44.3.5 Troubleshooting the Tags Service

This section provides troubleshooting tips on implementing the Tags service. It includes the following subsections:

- Section 44.3.5.1, "Tag Center Results Do Not Appear When Tag Is Clicked"
- Section 44.3.5.2, "The Tag Center Does Not Display Tagged Documents"
- Section 44.3.5.3, "The Tagging Button Does Not Display On Page"
- Section 44.3.5.4, "Clicking a Tagged Resource Does Not Go To The Correct Page"

44.3.5.1 Tag Center Results Do Not Appear When Tag Is Clicked

Problem

A resource to which a user has View permission has been tagged, but the user cannot see the resource in the Tag Center.

Solution

1. In `service-definition.xml`, verify that the resource type has the Resource Authorizer configured properly.
2. Enable logging for the Resource Authorizer to ensure that it is giving you permission to see the required objects.

44.3.5.2 The Tag Center Does Not Display Tagged Documents

Problem

Tagged WebCenter Portal Content Server documents do not appear in the Tag Center.

Solution

Check the version listed for the WebCenter Portal component in the Content Server. The result set name changed in WebCenter Portal release 11.1.1.6.0.

44.3.5.3 The Tagging Button Does Not Display On Page

Problem

The Tagging Button component is dropped on a page, but it does not show up.

Solution

- The Tagging Button requires the Tagging Dialog task flow to be present on the page. Drop this task flow on the page or page template.
- The Tagging Button requires a valid database connection. Ensure that there is a database connection present.

44.3.5.4 Clicking a Tagged Resource Does Not Go To The Correct Page

Problem

When clicking a tagging resource in Tag Center, users are not redirected to the correct page.

Solution

All tagging links are generated through the Resource Action Handling (RAH) framework. Ensure that RAH is configured properly.

- Check `service-definition.xml` for that resource type to see if a resource viewer is configured properly.
- Ensure that the `resourceID` and `serviceID` parameters passed for this resource are correct.

> **See Also:** Section 7.3, "Extending Your Framework Application with Custom Components"

45 Integrating the Search Service

This chapter explains how to integrate the Search service in a WebCenter Portal: Framework application at design time.

For more information about managing and including search, see:

- "Managing Oracle SES Search in WebCenter Portal" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the Search Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

This chapter includes the following sections:

- Section 45.1, "Introduction to Search"
- Section 45.2, "Basic Configuration for the Search Service"
- Section 45.3, "Advanced Information for the Search Service"

45.1 Introduction to Search

WebCenter Portal's search provides a unified, extensible framework that enables the discovery of information through an intuitive user interface. It honors application security settings by returning only the results a user is authorized to view. Users can save their searches for reuse, including searches with complex search refinement conditions.

WebCenter Portal provides two ways of searching your application:

- Oracle Secure Enterprise Search (SES) adapter
- Oracle WebCenter Portal live search adapters

In addition, the Documents service provides its own search engine for file searches. This saves time and increases the relevancy of results by narrowing the scope of a search to files.

> **See Also:** Section 14.4, "Using Portal Navigation to Create a Sitemap" for information about integrating an external search engine in your application

45.1.1 Understanding the Search Service

With Framework applications, Oracle Secure Enterprise Search (SES) is set as the default and preferred search platform. Oracle SES provides unified ranking results for the following resources:

- Documents, including wikis and blogs

- Announcements and Discussions
- Spaces, lists, page content, and people resources in WebCenter Portal: Spaces

> **Note:** WebCenter Portal: Spaces applications include the Oracle SES crawler that indexes spaces, lists, pages, and people resources in Spaces. This crawler is not supported in Framework applications.

Oracle SES search is beneficial for the following reasons:

- Oracle SES search provides unified ranking results, with the most relevant items appearing first.
- Oracle SES search allows search of repositories outside of WebCenter Portal. Oracle SES results appear in the same result set as WebCenter Portal's search results.
- Oracle SES search supports the Search service REST APIs and data control for customizing your search interface.

45.1.1.1 Understanding Search with the Oracle SES Adapter

Supported Oracle SES versions include 10.1.8.4.x and later. Oracle strongly recommends using Oracle SES release 11.1.2.2. Search with Oracle SES requires additional configuration in Oracle SES Administration, Oracle WebCenter Content: Content Server, and Oracle WebCenter Portal's Discussion Server.

> **See Also:** Section 45.3.4, "Configuring Search with the Oracle SES Adapter"

45.1.1.2 Understanding Search with WebCenter Portal's Search Adapters

Although Oracle SES is the preferred search platform for best performance, you can manually override search with Oracle SES and have Framework applications search using WebCenter Portal's original search adapters. WebCenter Portal's search adapters span all enabled and searchable services, such as documents, tags, people, and pages.

> **See Also:** Section 45.3.5, "Configuring Search with WebCenter Portal's Search Adapters"

45.1.1.3 Click Actions on Search Results

If no Resource Action Handling is specified in the Framework application, then the default action when a search result is clicked is to render the search result in a new browser window. Also, results from services that provide resource viewers, such as Documents, Discussions or Announcements, open in their resource viewers. For more information about resource viewers and the Resource Action Handling framework, see Section 7.3, "Extending Your Framework Application with Custom Components."

45.1.2 What Happens at Runtime

At run time, users can perform global (application-wide) searches from the Search task flow or from the Search toolbar and refine and save searches.

For more information about WebCenter Portal's search at runtime, see *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

45.2 Basic Configuration for the Search Service

This section provides the steps for adding the Search service to your application. It includes the following subsections:

- Section 45.2.1, "Configuration Roadmap - Oracle SES"
- Section 45.2.2, "Setting up Connections for Oracle SES"
- Section 45.2.3, "Adding the Search Service at Design Time"
- Section 45.2.4, "Setting Security for the Search Service"

45.2.1 Configuration Roadmap - Oracle SES

Use the roadmap in this section as a guide through the configuration process.

> **Note:** If you override search with Oracle SES to use the original WebCenter Portal live search adapters, then no Oracle SES configuration is required. Skip ahead to section Section 45.2.3, "Adding the Search Service at Design Time."

Figure 45–1 and Table 45–1 provide an overview of the prerequisites and tasks required to get Oracle SES working in Framework applications.

Figure 45–1 Configuring Oracle SES for Framework Applications

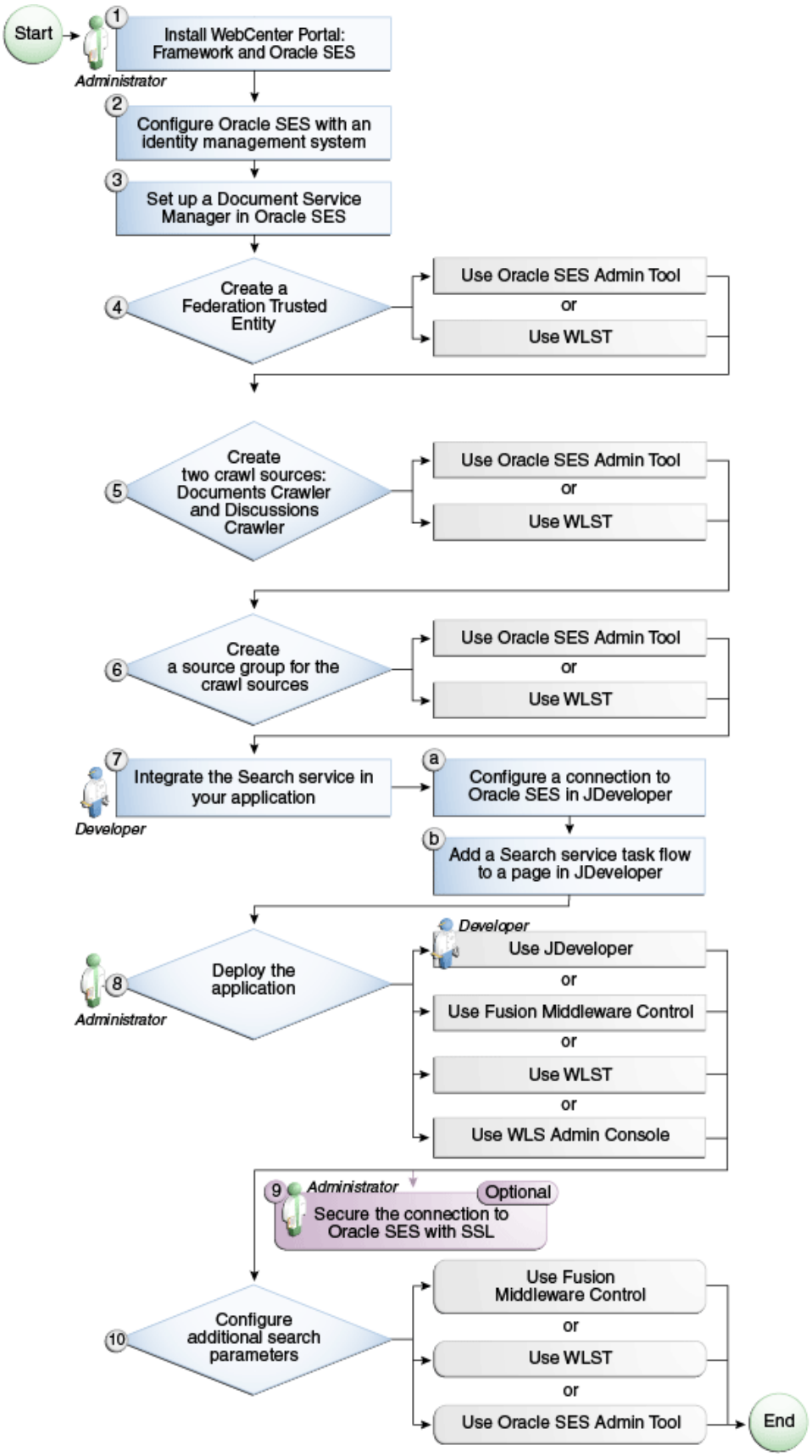


Table 45–1 Configuring Oracle SES for Framework Applications

Actor	Task	Sub-task
Administrator	1. Install WebCenter Portal and Oracle SES	
	2. Configure Oracle SES with an identity management system	
	3. Set up a Document Service Manager in Oracle SES	
	4. Create a Federation Trusted Entity using one of the following tools: ■ Oracle SES Admin Tool ■ WLST	
	5. Create two crawl sources: Documents Crawler and Discussions Crawler using one of the following tools: ■ Oracle SES Admin Tool ■ WLST	
	6. Create a source group for the crawl sources using one of the following tools: ■ Oracle SES Admin Tool ■ WLST	
Developer	7. Integrate the Search service in your application	7.a Configure a connection to Oracle SES in JDeveloper 7.b Add a Search service task flow to a page in JDeveloper
Developer or Administrator	8. Deploy the application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Console (Administrator)	
Administrator	9. (Optional) Secure the connection to Oracle SES with SSL	
Administrator	10. Configure additional search parameters using one of the following tools: ■ Fusion Middleware Control ■ WLST ■ WLS Admin Console	

45.2.2 Setting up Connections for Oracle SES

To search with Oracle SES, you must add the Oracle SES connection in the application. For more information, see Section 45.3.4, "Configuring Search with the Oracle SES Adapter."

> **Note:** You can create a connection to Oracle SES to search repositories outside WebCenter Portal and override using Oracle SES as the default search crawler with Fusion Middleware Control or WLST. This allows users to search other Framework application resources, such as people.

1. Add the search toolbar to a page, following Section 45.2.3, "Adding the Search Service at Design Time."
2. Set up a connection to Oracle SES. In the Application Navigator, expand **Application Resources**. Right-click the **Connections** node, select **New Connection**, and then **Oracle Secure Enterprise Search**.

   Select to create the connection in either the Application Resources or IDE connections. A connection in Application Resources is available only for that application, while a connection in IDE connections is available for all applications you create. If you plan to use the connection in other applications, then select IDE connections to avoid the need to re-create it.

   > **Note:** If you create the connection in IDE, then the connection must be added to the application later. For example, in the Resource Palette under IDE connections, right-click the connection and select **Add to Application**.

   Enter values for the fields in the dialog. For example:

   a. For **Connection Name**, enter a meaningful name, for example, `SESconnection`. This name cannot be changed after it is created. (To use a different connection name, create a new connection.)

   b. For **SOAP URL**, enter the URL for the Oracle SES Web Services endpoint, for example:

      `http://`*`host:port`*`/search/query/OracleSearch`

   c. Under **Federation Trusted Entity**, for **Entity Name**, enter the trusted entity defined for this application on the Oracle SES instance; for example, `wcsearch`.

      A trusted entity allows the application to authenticate itself to Oracle SES and assert its users when making queries on Oracle SES. This trusted entity can be any user that either exists on the identity management server behind Oracle SES or is created internally in Oracle SES. You can find (or create) the trusted entity on the **Global Settings - Federation Trusted Entities** page of Oracle SES.

   d. For **Entity Password**, enter the password of the Federation Trusted Entity user.

   e. Under **Application Configuration**, select the box for **Set as default connection for the Search service**.

      This ensures that your application uses that connection to access Oracle SES. The Search service uses only one Oracle SES connection.

Note: After you create a connection as the default connection, you cannot edit it so that it is *not* the default. To use a different default connection, you must create a new connection and mark that as the default connection.

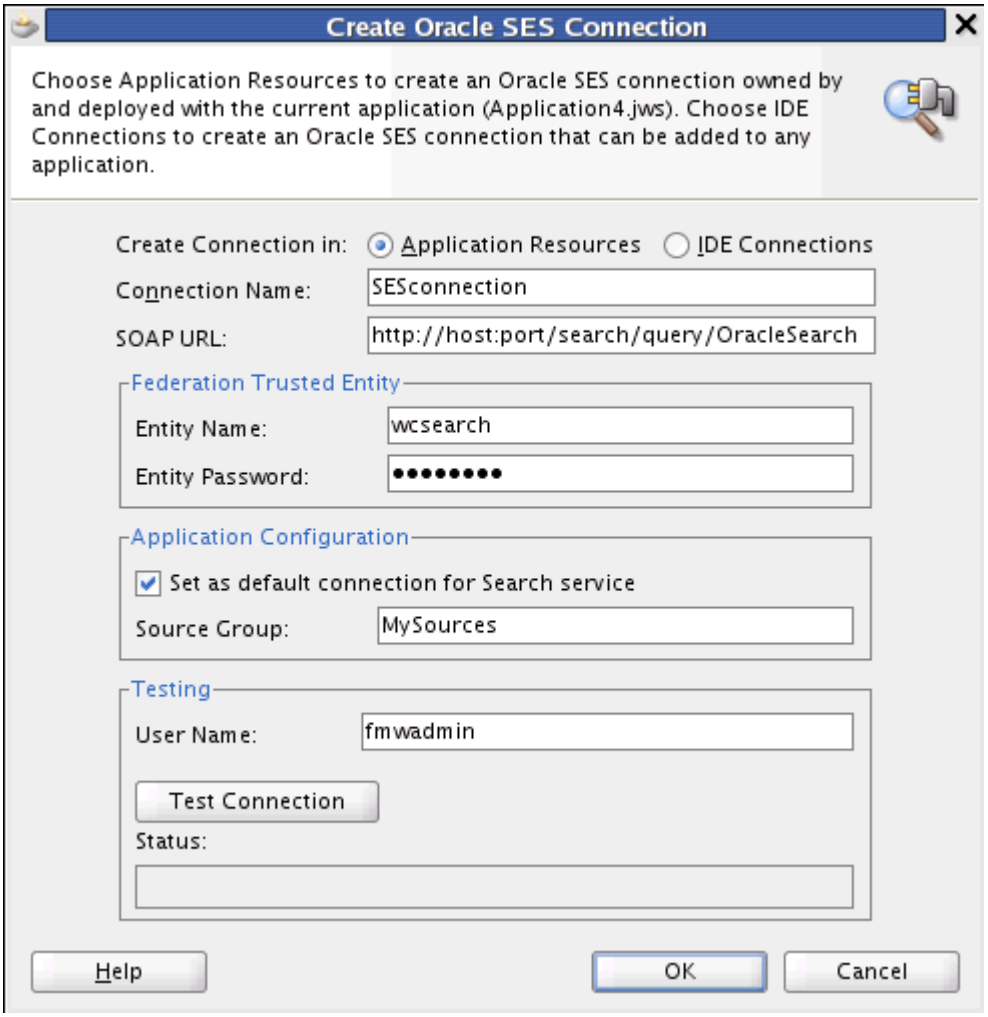
f. For **Source Group**, enter the Oracle SES source group in which to search. If a value is not provided, then all crawl sources in the Oracle SES instance are searched.

Note: Source groups present a good way to segment your searchable data on Oracle SES. Given an Oracle SES instance that keeps a search index on all corporate data, you can use source groups to make only a portion of your corporate data available to WebCenter Portal's search. For more information on creating source groups, see *Oracle SES Administrator's Guide*.

g. Under **Testing**, for **User name**, enter the name of a user present in both the Oracle Identity Management server configured for your application and the Oracle Identity Management server configured for Oracle SES.

You should see something similar to Figure 45–2.

Figure 45–2 Create Oracle SES Connection Dialog

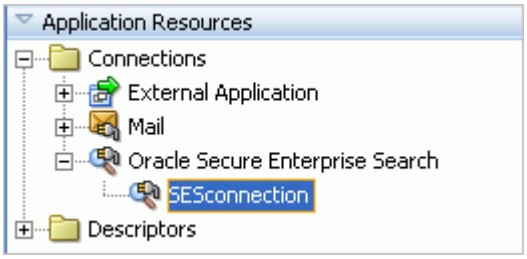


Click **Test Connection** and check the status.

If the connection is successful, then click **OK**.

After you have included the Oracle SES connection in your application, you should see it in your Application Resources panel, as shown in Figure 45–3.

Figure 45–3 Oracle SES Connection in Application Resources



The Oracle SES connection is now included in the `connections.xml` file and is the default connection in the `adf-config.xml` file, as shown in Example 45–1.

Example 45–1 Oracle SES Connection in adf-config.xml

```
<ses-properties>
  <connection>SESconnection</connection>
  <data-group>MySources</data-group>
</ses-properties>
```

> **Note:** After a Framework application has been deployed to an Oracle WebLogic managed server, you can configure it using WLST or Fusion Middleware Control. For more information, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

45.2.3 Adding the Search Service at Design Time

This section describes the Search service task flows and explains how to integrate the Search service into your application. It includes the following subsections:

- Section 45.2.3.1, "Search Service Task Flows"
- Section 45.2.3.2, "How to Add the Search Service to Your Page"
- Section 45.2.3.3, "How to Modify the Search Service Task Flow Parameters"

45.2.3.1 Search Service Task Flows

Table 45–2 lists the Search service task flows.

Table 45–2 Search Service Task Flows

Task Flow	Definition
Search	This task flow provides a rich search experience with features for refining and saving search results.
Search - Saved Searches	This task flow enables you to create a simple launch pad for running saved searches within the application.
Search Preferences	This task flow enables users to select which WebCenter Portal services to search.
Search Toolbar	This task flow enables users to enter simple search criteria and run the search from the application. Search results are rendered as links.

45.2.3.2 How to Add the Search Service to Your Page

The Search Toolbar task flow offers the easiest way to add the Search service to your application. To add the Search Toolbar task flow to your Framework application:

1. Follow the steps described in Chapter 3, "Preparing Your Development Environment" to create a customizable page in your application.

> **Note:** ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

2. Open the page on which to add the Search service.
3. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.

4. Drag and drop the **Search Toolbar** task flow onto the page.
5. When prompted, select **Region** as the way to create the task flow.

   If a prompt to confirm the ADF library appears, click **Add Library**.
6. Run your page with the search toolbar on it.

   In the Application Navigator, right-click this application and select **Run**. It may take a few moments for the operation to complete.

   Enter the user name and credentials that you created when you defined security.

> **Note:** To return results, the Search service requires that other WebCenter Portal services are included in the application.

45.2.3.3 How to Modify the Search Service Task Flow Parameters

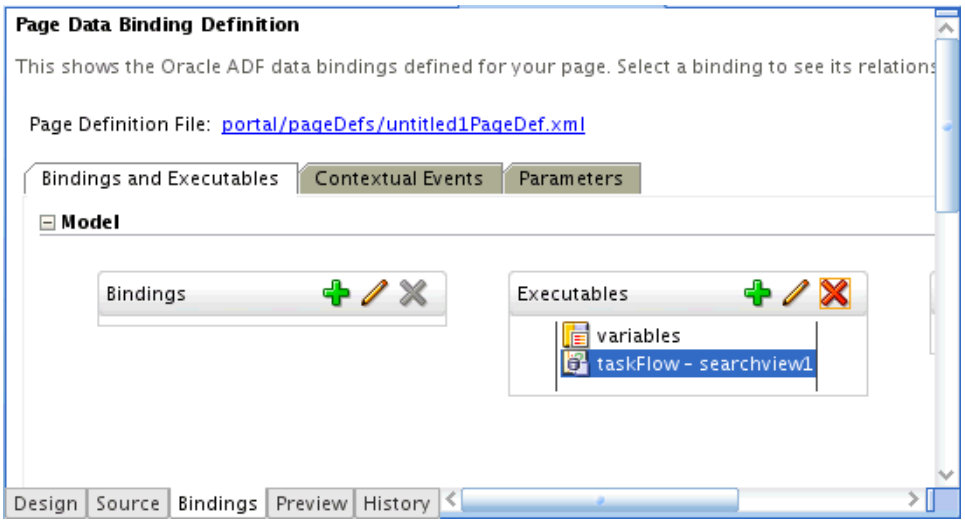
With the Search service task flow parameters, you can customize the appearance and behavior of a task flow instance. For example, you can restrict search results to include only specific services, document types, and spaces, or you can change the size of the search box.

> **See Also:** "How to Set Parameters to Narrow Search" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* for detailed information about customizing the Search service with task flow parameters

To see and adjust task flow binding parameter values after you have placed a task flow on a page:

1. Navigate to the Edit Task Flow Binding dialog by clicking the **Bindings** tab at the bottom of the page containing the task flow (next to the **Source** tab).
2. Under **Executables**, you'll see the task flow you added. Figure 45–4 shows an example of a Search task flow in the Executables section.

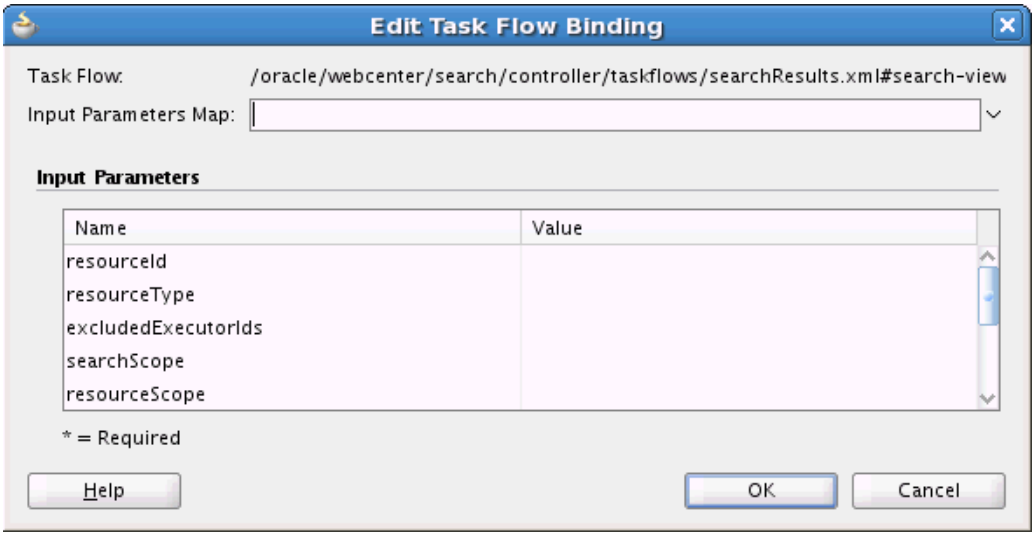
Figure 45–4 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.

4. In the Edit Task Flow Binding dialog Figure 45–5, revise the binding parameter value by double-clicking the field and either changing the existing value or adding a new one.

Figure 45–5 Edit Task Flow Binding Dialog for Search Task Flow



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

Table 45–3 describes the properties that are unique to the Search service task flows.

Table 45–3 Search Service Task Flow Parameters

Property	Description	Task Flow
`customAttributes`	List of custom attributes to show when displaying search results. To include one or more custom attributes in the search results, set this to a list of custom attribute names, separated by commas. An optional label prefix may be provided with the custom attribute name to display instead of its associated custom attribute name. Use the format: label:name.	Search Search Toolbar
`hideRefiners`	List of refiners to hide when displaying search results. To hide one or more refiners, set this to a list of refiner names, separated by commas (choose from creator, date, space, content, and tags).	Search Search Toolbar
`keywordsInputRendered`	Show or hide the input box. Set to true (default) to show the input box, or set to false to hide it.	Search

Table 45–3 (Cont.) Search Service Task Flow Parameters

Property	Description	Task Flow
`mimeType`	List of MIME types to limit the search. To limit the search to certain document types, set this to the list of MIME types of the documents (such as PDF, PPT, DOC), separated by commas. This also can be set to the Documents service with oracle.webcenter.doclib. **Note**: To limit search to Microsoft Word documents and WebCenter Portal pages, set this parameter to application/msword, and set the `serviceIds` parameter to oracle.webcenter.doclib,oracle.webcenter.page. **See Also**: "Searching Specific Document Types" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*	Search Search Toolbar
`searchScope`	List of unique IDs to limit the search scope. To limit the search to a list of spaces, set this to the list of GUIDs of the spaces, separated by commas. For example: `#{spaceContext.currentSpaceGUID}` or `s8bba98ff_4cbb_40b8_beee_296c916a23ed`	Search Search Toolbar
`resourceScope`	Unique ID to limit the search scope. To limit the search to a particular space, set this to the GUID of the space. For example: #{spaceContext.currentSpaceGUID} or s8bba98ff_4cbb_40b8_beee_296c916a23ed **Note:** This parameter has been deprecated. It is included for backward-compatibility only.	Search
`searchBoxSize`	Value to limit the size of the search box. The default value is 42. Enter a lower number (for example, 30) to shorten the length of the search box. This also changes the size of the search box in the Search task flow.	Search Toolbar
`serviceIds`	List of IDs of services or executors to include when displaying search results. To display only people profiles and documents in the Content Server, set this parameter to `oracle.webcenter.peopleconnections.profile, oracle.webcenter.doclib`. If nothing is specified, then all services are searched. For a list of service IDs, see Table G–7, " Service IDs".	Search Search Toolbar
`resourceId`	Either search keywords or the saved search GUID. **Note:** This parameter is for internal use only. Do not change this value unless you want coded search main views. If you do change this value, then you must also specify `resourceType`.	Search

Table 45–3 (Cont.) Search Service Task Flow Parameters

Property	Description	Task Flow
`resourceType`	Marker specifying whether the `resourceId` parameter searches keywords or saved search GUID. **Note:** This value is set automatically and is for internal use only. Do not change it unless you want coded search main views.	Search
`scope`	List of unique objects to limit the search scope. **Note:** This parameter has been deprecated. Use the `searchScope` parameter instead. (If a value is set for this parameter, then any value in `searchScope` is ignored.)	Search Toolbar

Note: The All Saved Searches and the Search Preferences task flows do not have any unique properties.

45.2.4 Setting Security for the Search Service

The Oracle SES adapter must be configured with an identity management system to validate and authenticate users. This is necessary for searches to return only results that the user is allowed to view based on access privileges. Because WebCenter Portal uses identity propagation when communicating with Oracle SES, WebCenter Portal's user base must match that in Oracle SES. One way this can happen is by configuring WebCenter Portal and Oracle SES to the same identity management system. For detailed information, see the section "Oracle SES - Configuration" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

WebCenter Portal's original search adapters rely on individual services to ensure that the user performing the search can view results. Search results are returned based on user privileges. When unauthenticated, queries return only public content. Only authenticated users of an ADF-secured Framework application can save searches.

ADF security is configured by default if you created your application using WebCenter Portal's Framework application template. For information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

45.3 Advanced Information for the Search Service

This section describes optional features available with this service. It includes the following subsections:

- Section 45.3.1, "Adding the Search Task Flow"
- Section 45.3.2, "Adding the Search - Saved Searches Task Flow"
- Section 45.3.3, "Adding the Search Preferences Task Flow"
- Section 45.3.4, "Configuring Search with the Oracle SES Adapter"
- Section 45.3.5, "Configuring Search with WebCenter Portal's Search Adapters"
- Section 45.3.6, "Using the Search Service Java APIs"
- Section 45.3.7, "Using the Search Service REST APIs"
- Section 45.3.8, "Using the Search Service Data Control"

- Section 45.3.9, "Building Adapters for the Search Service"
- Section 45.3.10, "Customizing the Search UI Without WebCenter Portal's Search Service APIs"
- Section 45.3.11, "Troubleshooting the Search Service"

> **Note:** The Search service REST APIs and data controls are available only for search using Oracle SES. They are not supported with search using WebCenter Portal's original search adapters.

45.3.1 Adding the Search Task Flow

To add the Search task flow to your application, follow the instructions provided for the Search Toolbar task flow in Section 45.2.3, "Adding the Search Service at Design Time," but instead, drag and drop the Search task flow onto the page.

To run the page with the Search view:

1. In the Application Navigator, right-click the page and select **Run**. As long as the application has been ADF-secured, you should see a login page. It may take a few moments for the operation to complete.
2. If you configured ADF security, then enter the user name and password that you provided and click **Submit**.
3. Enter some search criteria in the field and click the **Search** icon. You may get few, if any, results if the application is not populated with much besides the Search service itself.

45.3.2 Adding the Search - Saved Searches Task Flow

You can add the Search - Saved Searches task flow to provide a simple launch pad for running saved searches within the application. To add this task flow to your Framework application, follow the instructions provided for the Search Toolbar task flow in Section 45.2.3, "Adding the Search Service at Design Time," but instead, drag and drop the Search - Saved Searches task flow onto the page.

The task flow renders links that, when clicked, run a saved search.

45.3.3 Adding the Search Preferences Task Flow

The Search Preferences task flow enables users to control which WebCenter Portal services are searched. By default, all enabled services are selected to be searched. Users can disable any service from which they do not want to see search results. Users also can select the order of services in their search results. For example, they may find that search results from the Documents service prove more useful that search results from other services.

To add the Search Preferences task flow to your application, follow the instructions provided for the Search Toolbar task flow in Section 45.2.3, "Adding the Search Service at Design Time," but instead, drag and drop the Search Preferences task flow onto the page.

45.3.4 Configuring Search with the Oracle SES Adapter

WebCenter Portal: Framework applications automatically set Oracle Secure Enterprise Search (SES) as the default search adapter, but additional configuration is required in Oracle SES, Oracle WebCenter Content: Content Server, and Oracle WebCenter Portal's

Discussion Server. See the section "Managing the Search Service" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* for detailed information about the steps to set up search with the Oracle SES adapter.

After you have completed all configuration steps, validate the inclusion of Oracle SES results in WebCenter Portal's search.

1. In the Application Navigator, right-click the page you created with the Search Toolbar, and click **Run**.

   As long as the application has been ADF-secured, you should see a login page. It may take a few moments for the operation to complete.

2. Enter the user name and credentials that you created when you defined security.

3. Click **Submit**.

   When the page loads, you should see your search toolbar.

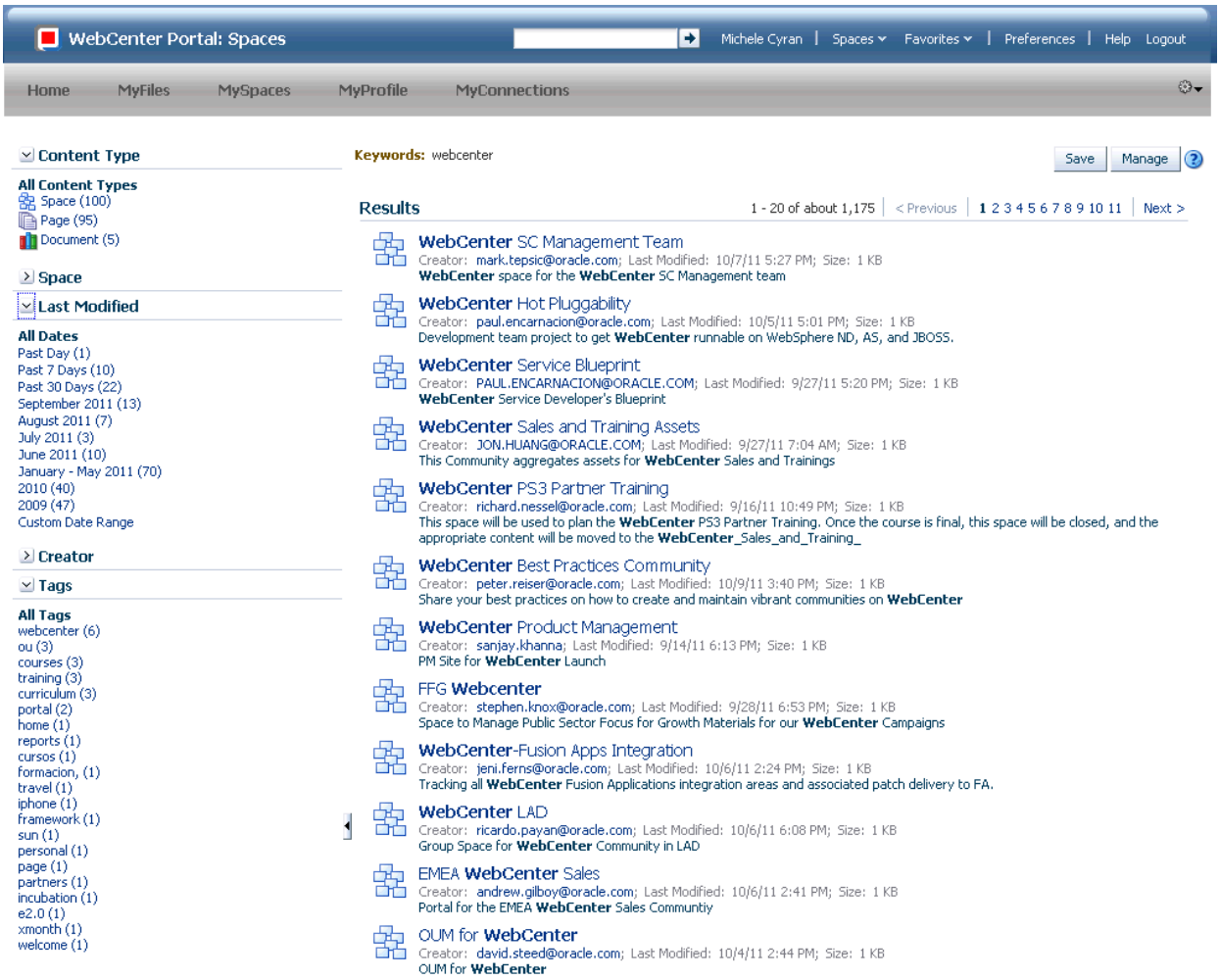
   Figure 45–6 shows a sample results page for the search term *webcenter*, before Oracle SES has crawled anything.

Figure 45–6 Search Results



Figure 45–7 shows a sample results page after Oracle SES has been configured and crawled.

Figure 45–7 Global Search Results



See Also: Oracle SES documentation provided on the Oracle Fusion Middleware documentation library (in WebCenter Portal's product area)

45.3.5 Configuring Search with WebCenter Portal's Search Adapters

You can manually override search with Oracle SES and have Framework applications search using WebCenter Portal's original search adapters. WebCenter Portal's search adapters span all enabled and searchable services, such as documents, tags, people, and pages.

To enable search with WebCenter Portal's search adapters, edit the `adf-config.xml` file either to change all enable attributes in `crawl-properties` to false (Example 45–2) or to comment out the `crawl-properties` entirely (Example 45–3).

Example 45–2 Crawl Properties in adf-config.xml Set to False

```
  <searchC:adf-search-config
xmlns="http://xmlns.oracle.com/webcenter/search/config">
    <display-properties>
      <common numSavedSearches="5"/>
      <region-specific>
        <usage id="simpleSearchResultUIMetadata" numServiceRows="5"/>
```

```
        <usage id="searchResultUIMetadata" numServiceRows="5"/>
        <usage id="localToolbarRegion" numServiceRows="5"/>
      </region-specific>
    </display-properties>
    <execution-properties timeoutMs="7000" prepareTimeoutMs="3000"/>
    <crawl-properties fullCrawlInterval="P5D" enableWcServicesCrawl="false"
                      enableWcDiscussionsCrawl="false" enableWcUcmCrawl="false"/>
    <ses-properties>
      <connection/>
      <data-group/>
    </ses-properties>
  </searchC:adf-search-config>
```

Example 45–3 Crawl Properties in adf-config.xml Commented Out

```
  <searchC:adf-search-config
xmlns="http://xmlns.oracle.com/webcenter/search/config">
    <display-properties>
      <common numSavedSearches="5"/>
      <region-specific>
        <usage id="simpleSearchResultUIMetadata" numServiceRows="5"/>
        <usage id="searchResultUIMetadata" numServiceRows="5"/>
        <usage id="localToolbarRegion" numServiceRows="5"/>
      </region-specific>
    </display-properties>
    <execution-properties timeoutMs="7000" prepareTimeoutMs="3000"/>
    <!--
    <crawl-properties fullCrawlInterval="P5D" enableWcServicesCrawl="true"
                      enableWcDiscussionsCrawl="true" enableWcUcmCrawl="true"/>
    -->
    <ses-properties>
      <connection/>
      <data-group/>
    </ses-properties>
  </searchC:adf-search-config>
```

45.3.6 Using the Search Service Java APIs

Custom components can implement Search service APIs to expose their content to WebCenter Portal's search.

For more information, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

45.3.7 Using the Search Service REST APIs

WebCenter Portal provides REST APIs to support the Search service. Use the Search service REST APIs to post, read, update, and delete searches. You can specify keywords and the scope of the search; for example, the iPhone could search for "smith" in all spaces, documents and wiki pages. The API also lets you build a customized search user interface on top of the Search service.

This section describes the REST APIs associated with the Search service. It includes the following subsections:

- Section 45.3.7.1, "Search Entry Point"
- Section 45.3.7.2, "Search Resource Type Taxonomy"
- Section 45.3.7.3, "Security Considerations"

- Section 45.3.7.4, "Search Resource Types"

For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

45.3.7.1 Search Entry Point

Each REST service has a link element within the resource index that provides the entry point for that service. To find the entry point for the Search service, find the link element with a `resourceType` of:

```
urn:oracle:webcenter:search
```

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Search service.

For more information about the resource index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

45.3.7.2 Search Resource Type Taxonomy

When the client has identified the entry point, it then can navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 54.5.2.1, "Resource Type."

The taxonomy for the Search service is:

```
urn:oracle:webcenter:search
   urn:oracle:webcenter:search:results
```

Beyond the service entry points, URL templates allow clients to pass query parameters to customize their requests and control the form of returned data.

Collection resources in the search resources support pagination (`startIndex` and `itemsPerPage`). Other query parameters are not supported (that is, `search` and `projection`).

45.3.7.3 Security Considerations

Authentication is required before using Search REST API methods.

For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

45.3.7.4 Search Resource Types

This section provides information about each resource type. It includes the following subsection:

- Section 45.3.7.4.1, "urn:oracle:webcenter:search:results"

45.3.7.4.1 urn:oracle:webcenter:search:results Use this resource type to identify the URI to use to read (`GET`) a query containing keywords and attribute-specific criteria.

The request is represented by the URL and the response is a list of search results, each with metadata to help the end user choose an item to drill down, as well as cross links to the owning REST services, if available. If the owning REST service is unavailable, then an HREF link is provided.

The response XML can be paginated with standard URL parameters, and appropriate previous and next links provided along with a general template (with the query prepopulated) for the consuming application to do its own custom pagination.

Navigation Paths to results

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   search

resourceindex
   spaces
      spaces:resourceindex
         spaces:search
```

Supported Methods for results

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters**: `serviceId`, `q`, `scopeGuid`, `refiners`, `startIndex` (pagination), `itemsPerPage` (pagination)
  - **response - body:** search results, and refiners (if `refiners=true`)

where:

- `serviceId={serviceId1}[,{serviceId2}...]`

  Optional: This narrows the search to one or more WebCenter Portal services, such as oracle.webcenter.doclib or oracle.webcenter.page. For example, setting this to oracle.webcenter.doclib searches only documents. If omitted, this searches all services.

  See Also: Table G–7, " Service IDs" for a list of service IDs

- `q={queryString}`

  Searches may be specified using the following format

  `[[`*`field1`*`:[`*`operand`*`]][:]`*`value1`*`[;`*`field2`*`:`*`operand`*`:`*`value2`*`]]`

  Where multiple clauses are joined by an implicit `AND` and are syntactically separated by the ";" semicolon. Square brackets `[]` denote optional values. Each clause can be one of the following types:

  - `keywords` - simple string with no ":" colon separator
  - `attribute1:value1` - shortcut for a clause meaning attribute1 equals value1
  - `attribute1:operand1:value1` - a clause capturing a Boolean condition between attribute1 and value1

  Search attributes include the following, but can be *any* attribute name:

  - `title` - Name or title of the item
  - `description` - Description of the item
  - `creator` - Creator or contributors of the item
  - `date` - Date on which the item was last modified

  - `tags` - Tag words that the item may have (applicable to pages)

  The attributes can be returned in search results, where `data=field1[,field2]`

  Legal operands:

  - `equals` - Strings and Numbers
  - `contains` - Strings
  - `gt` (greater than) - Numbers
  - `gte` (greater than or equals) - Numbers
  - `lt` (less than) - Numbers
  - `lte` (less than or equals) - Numbers

- `scopeGuid={...}`

  Optional: The GUID associated with the scope.

- `refiners={true|false}`

  Optional: Whether to fetch refiners. If omitted, the default value is false.

Example 45–4 Search REST Commands

```
http://examplehost:8888/rest/api/searchresults?q=document&serviceId=oracle.webcenter.doclib,oracle.webcenter.page&utoken={utoken}

http://examplehost:8888/rest/api/searchresults?q={keywords};dDocName:contains:handbook&data=dDocName,dOriginalName
```

For more information, see Section 54.5.2.5, "Templates."

Writable Elements for results

Table 45–4 lists the writable elements for this resource.

Table 45–4 Writable Elements for results

Element	Type	Required	Constraints	Description
`author`	String		1 or more characters	Name of the query author
`resourceId`				
`serviceId`				
`title`				
`description`	String	No	1 or more characters	Description of the query
`lastModifiedDate`				
`size`				

Resource Types Linked to From results

Table 45–5 lists the resource types that the client can link to from this resource.

Table 45–5 Related Resource Types for search

rel	resourceType
self	

45.3.8 Using the Search Service Data Control

Use the Search service data control to build a customized search user interface with a Framework application or a custom task flow. Deploying this task flow into an ADF library allows for a portable consumption of the task flow; for example, you could add it to a WebCenter Portal: Spaces application Resource Catalog to build site templates.

45.3.8.1 Search Data Control Methods, Attributes, and Classes

The Search data control contains the `searchSes` method. Table 45–6 describes its parameters for setting search refiners and limiting the scope of the search.

Table 45–6 searchSes Input Parameters

Parameter	Required?	Type	Description
`scopeGuid`	No	String	GUID value to filter results by spaces.
`serviceId`	No	String	The service ID of the item; for example, oracle.webcenter.doclib searches only documents. If omitted, this searches across all services.
`keywords`	Yes	String	Search terms.
`predicates`	No	List	Collection of predicates.
`fetchRefiners`	No	Boolean	Whether to fetch refiners. If omitted, the default value is false.
`startIndex`	Yes	Integer	Specifies the index of the first matching result that should be included in the result set (0-n ... zero based). This is used for pagination.
`itemsPerPage`	Yes	Integer	Number of results to display.

The `searchSes` method returns:

- Collection of SearchServiceDCRows objects
- Collection of SearchServiceDCRefiner objects

45.3.8.1.1 SearchServiceDCRows

- author
- comments
- createDate
- description
- iconPath
- language
- largeIconPath
- lastModifiedDate
- resourceId
- scopeGuid
- serviceId
- size
- tagWords

- title
- type
- urlSearchServiceDCRefiner

45.3.8.1.2 SearchServiceDCRefiner

element

45.3.8.1.3 SearchServiceDCRefinement

- keywords
- scopeGuid
- serviceId
- Collection of predicates

45.3.8.2 Integrating the Search Data Control

To use the Search data control:

1. In JDeveloper, create a Framework application with an Oracle SES connection. After the connection is available, the Search Data Control is listed in the Data Controls palette.
2. Drag and drop the `searchSes` method from `SearchServiceDC`.
3. Choose **ADF Parameter Form**, and the Edit Form Fields dialog opens. Click **OK**.
4. Drag the rows accessor, from the `SearchServiceDCResult` return object type, onto the form, and choose **Table - ADF Read-only Table**. Click **OK**.
5. Run the page, enter appropriate values. For example:
   - `keywords`: any search term
   - `startIndex`: 0
   - `itemsPerPage`: 10

For information about adding data controls to your application, see Section 7.1.2, "Understanding WebCenter Portal APIs."

45.3.9 Building Adapters for the Search Service

You can use search adapters to add other sources to your search results. WebCenter Portal framework automatically discovers these search adapters and consolidates them in formulating search results in your application. Subsequently, when you expose your custom component to the Search service in your application, it is included in the federated search.

This section describes how to extend WebCenter Portal's search through search adapters. It includes the following subsections:

- Section 45.3.9.1, "How to Add a Search Source"
- Section 45.3.9.2, "How to Register a Custom Adapter"
- Section 45.3.9.3, "Search Adapter Attributes"
- Section 45.3.9.4, "What Happens at Runtime"

45.3.9.1 How to Add a Search Source

To add a new search source, you must create Java classes for the new source to manage the incoming queries and return search results. After you add the necessary Java classes to your application, you must register them with the Search service.

When performing a search, the search framework calls `QueryManager.createRowQueryHandler` with a query object and a `List<QName>` that describes the columns sought. In return, the framework receives a `QueryHandler<Row>`.

A `QueryHandler` can be a `QueryFederator` or a `QueryExecutor`. A `QueryFederator` is just a collection of other `QueryHandlers`. With the `QueryFederator`, the framework can get a list of the child `QueryHandlers` and keep recursing until all `QueryExecutors` are found.

The framework calls the `execute` method on the `QueryExecutor` to get a `QueryResult<Row>`, which is an extension of `java.util.Iterator<Row>` that iterates and retrieves rows of results.

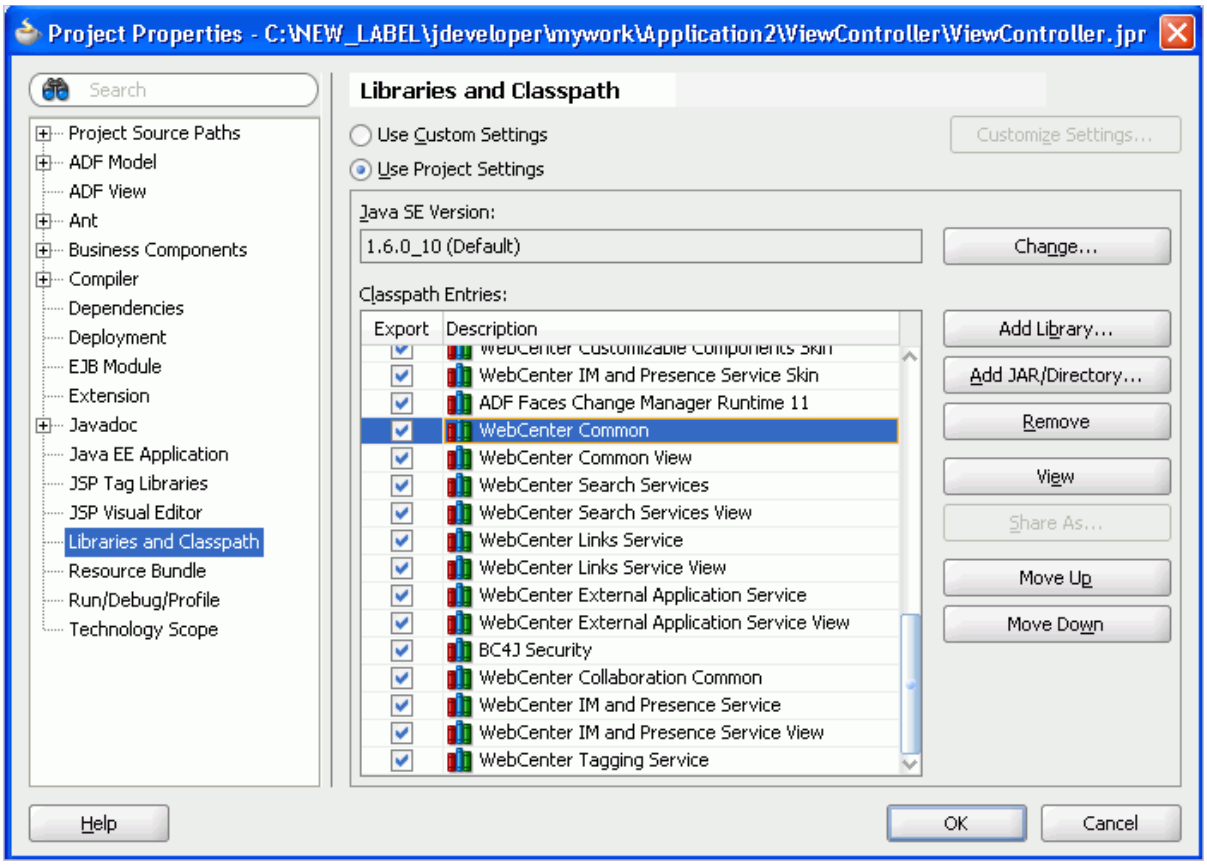
One possible implementation would be to create these classes:

- `SampleQueryManager` ...
- `SampleRowQueryExecutor` ...
- `SampleRowQueryResult` ...
- `SampleRow` ...

To add a new search source with a similar query manager:

1. In Oracle JDeveloper, open the application in which to add the search source.
2. Confirm that the WebCenter Common library is included in your project:
   a. In the Application Navigator, right-click your project and select **Project Properties** and then **Libraries and Classpath**.
   b. If the **WebCenter Common** library is not included, as shown in Figure 45–8, then click **Add Library** and select it.

Figure 45–8 WebCenter Common Library



c. Click **OK**.

3. Create a class called `SampleQueryManager.java` (Example 45–5).

Example 45–5 SampleQueryManager.java

```
package mycompany.myproduct.myapp.mycomponent.query;
import java.util.List;
import oracle.webcenter.search.QName;
import oracle.webcenter.search.Query;
import oracle.webcenter.search.QueryExecutor;
import oracle.webcenter.search.QueryManager;
import oracle.webcenter.search.Row;
public class SampleQueryManager
  implements QueryManager
{
  public SampleQueryManager()
  {
  }
  public QueryExecutor<Row> createRowQueryHandler(Query query,
                                                  List<QName> columns)
  {
    return new SampleRowQueryExecutor(query, columns);
  }
}
```

4. Create a class called `SampleRowQueryExecutor.java` (Example 45–6).

Example 45–6 SampleRowQueryExecutor.java

```
package mycompany.myproduct.myapp.mycomponent.query;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import static oracle.webcenter.search.AttributeConstants.DCMI_EXTENT;
import static oracle.webcenter.search.AttributeConstants.DCMI_MODIFIED;
import static oracle.webcenter.search.AttributeConstants.DCMI_CREATOR;
import static oracle.webcenter.search.AttributeConstants.DCMI_TITLE;
import static oracle.webcenter.search.AttributeConstants.DCMI_URI;
import static oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER;
import static oracle.webcenter.search.AttributeConstants.WPSAPI_ICON_PATH;
import oracle.webcenter.search.QName;
import oracle.webcenter.search.Query;
import oracle.webcenter.search.QueryExecutor;
import oracle.webcenter.search.QueryHandler;
import oracle.webcenter.search.QueryResult;
import oracle.webcenter.search.Row;
public class SampleRowQueryExecutor
  implements QueryExecutor<Row>
{
  private static final int PRESET_ESTIMATED_RESULT_COUNT = 10;
  private static final int PRESET_VALUES_BATCH_SIZE = 5;
  private static final int MAX_RESULT_LIMIT = 100;
  private int m_resultStart;
  private int m_resultLimit;
  private Query m_query;
  private List<QName> m_columns;
  private Map<String, String> m_properties;
  public SampleRowQueryExecutor(Query query, List<QName> columns)
  {
    m_query = query;
    m_columns = columns;
    m_properties = new HashMap<String, String>();
    // In actuality you may want to make sure this string is translatable
    m_properties.put(QueryHandler.TITLE, "Sample");
  }
  /**
   * Create several rows for "batchRepeat" times.
   */
  private static void createRows(List<Row> rows, int batchRepeat)
  {
    while (batchRepeat-- > 0)
    {
      rows.add(new SampleRow("23847", "Initial Sorting of Tables",
                             "Bar", "2006-06-08", 5120));
      rows.add(new SampleRow("4827445", "Support for facelets?",
                             "Foo", "2006-04-17", 1400000));
      rows.add(new SampleRow("952787", "For Thomas Kincade's validation demo",
                             "Tiger", "2006-01-09", 1300000));
      rows.add(new SampleRow("4394588", "Page not showing in WYSIWYG fashion",
                             "Ken Swartz", "2006-04-27", 3000));
      rows.add(new SampleRow("3920", "Tree table",
                             "Scott", "2006-03-24", 1200));
    }
```

```
  }
  /**
   * Instead of doing a query execution here,
   * just create a bunch of rows.
   */
  public QueryResult<Row> execute()
  {
    List<Row> rows = new ArrayList<Row>();
    // To hit a certain count here but not go over the max
    int realLimit = Math.min(MAX_RESULT_LIMIT, m_resultLimit);
    createRows(rows, (realLimit / PRESET_VALUES_BATCH_SIZE));
    // Create the QueryResult
    QueryResult<Row> ret = new SampleRowQueryResult(rows.iterator());
    ret.getProperties().put(QueryResult.ESTIMATED_RESULT_COUNT,
                            Integer.toString(PRESET_ESTIMATED_RESULT_COUNT));
    return ret;
  }
  /**
   * Remember the result start.
   */
  public void setResultStart(int resultStart)
  {
    m_resultStart = resultStart;
  }
  /**
   * Store away the result limit to use in the execute
   * to only get that number of results back.
   */
  public void setResultLimit(int resultLimit)
  {
    m_resultLimit = resultLimit;
  }
  /**
   * Expose our properties map containing our title.
   */
  public Map<String, String> getProperties()
  {
    return m_properties;
  }
}
```

5. Create a class called `SampleRowQueryResult.java` (Example 45–7).

Example 45–7 SampleRowQueryResult.java

```
package mycompany.myproduct.myapp.mycomponent.query;
import java.util.Iterator;
import oracle.webcenter.search.util.WrapperQueryResult;
import oracle.webcenter.search.Row;
public class SampleRowQueryResult extends WrapperQueryResult<Row>
{
  public SampleRowQueryResult(Iterator<Row> rows)
  {
    super(rows);
  }
}
```

6. Create a class called `SampleRow.java` (Example 45–8).

Example 45–8 SampleRow.java

```
package mycompany.myproduct.myapp.mycomponent.query;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import oracle.webcenter.search.QName;
import oracle.webcenter.search.Row;
import static oracle.webcenter.search.AttributeConstants.DCMI_EXTENT;
import static oracle.webcenter.search.AttributeConstants.DCMI_MODIFIED;
import static oracle.webcenter.search.AttributeConstants.DCMI_CREATOR;
import static oracle.webcenter.search.AttributeConstants.DCMI_TITLE;
import static oracle.webcenter.search.AttributeConstants.DCMI_URI;
import static oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER;
import static oracle.webcenter.search.AttributeConstants.WPSAPI_ICON_PATH;
public class SampleRow implements Row
{
  private Map<QName, Object> m_storage = new HashMap<QName, Object>();
  private DateFormat m_dateFormat = new SimpleDateFormat("yyyy-MM-dd");
  SampleRow(String id, String title,
            String creator, String lastModified,
            int size)
  {
    m_storage.put(DCMI_IDENTIFIER, id);
    // Only if you have an external URL to go to
    // m_storage.put(DCMI_URI, "http://my.external.url");
    m_storage.put(DCMI_TITLE, title);
    m_storage.put(DCMI_CREATOR, creator);
    // The below path points to a path accessible in the classpath to an icon
    m_storage.put(WPSAPI_ICON_PATH, "/adf/webcenter/search_qualifier.png");
    try
    {
      Date date = m_dateFormat.parse(lastModified);
      Calendar cal = Calendar.getInstance();
      cal.setTime(date);
      m_storage.put(DCMI_MODIFIED, cal);
    }
    catch (ParseException e)
    {
      e.printStackTrace();
    }
    m_storage.put(DCMI_EXTENT, new Long(size));
  }
  public Object getObject(QName columnName)
  {
    return m_storage.get(columnName);
  }
  public Iterator<QName> getColumns()
  {
    return m_storage.keySet().iterator();
  }
}
```

45.3.9.2 How to Register a Custom Adapter

To register your query manager with the Search service, you must edit the `service-definition.xml` file. The `SampleQueryManager` line exposes the custom adapter's query manager implementation to the Search service so that the custom adapter is included with any search user interface.

The entry should be similar to Example 45–9.

Example 45–9 Sample Query Manager Entry in service-definition.xml

```
<service-definition id="mycompany.myproduct.myapp.mycomponent"
   version="11.1.1.0.0">
  <resource-view taskFlowId="/somepath/somefile.xml#someId"
                 authorizerClass="some.service.id.Authorizer1"/>
  <name>Custom Component 1</name>
  <description>Description for Custom Component 1</description>
  <search-definition xmlns="http://xmlns.oracle.com/webcenter/search"
      id="mycompany.myproduct.myapp.mycomponent.query" version="11.1.1.0.0">
  <query-manager-class>
mycompany.myproduct.myapp.mycomponent.query.SampleQueryManager
  </query-manager-class>
  </search-definition>
</service-definition>
```

For more information about the `resource-view` tag (and WebCenter Portal's Resource Action Handling framework), see Section 7.3, "Extending Your Framework Application with Custom Components."

45.3.9.3 Search Adapter Attributes

This section describes the attributes available for searching a service. It includes the following subsections:

- Section 45.3.9.3.1, "Searchable Attributes"
- Section 45.3.9.3.2, "Keywords"
- Section 45.3.9.3.3, "Date Condition"
- Section 45.3.9.3.4, "Complex Date Condition"
- Section 45.3.9.3.5, "Person Condition"
- Section 45.3.9.3.6, "Complex Person Condition"
- Section 45.3.9.3.7, "Access"
- Section 45.3.9.3.8, "Selectable Attributes"

45.3.9.3.1 Searchable Attributes WebCenter Portal's search user interface exposes searches based on three fields: keywords, date, and person. Each service, in its own search implementation, honors these fields to the best of its capabilities. At minimum, it supports the keywords field. When you define your own search adapter, you must honor these fields to the best of your adapter's capabilities, as described in Section 45.3.9.3.8, "Selectable Attributes."

45.3.9.3.2 Keywords By default, the keyword field is in `oracle.webcenter.search.TextPredicate`. The field can comprise multiple words, with the service determining how to interpret multiple words. The Search service does not tokenize before the call comes to the service, but here are some guidelines:

- If two or more tokens are enclosed within quotes, then they should be interpreted as one word.
- Unless the `OR` operator is explicitly set, it is assumed that the default conjunction among the words in the keywords field is `AND`.

45.3.9.3.3 Date Condition For Search service to support filtering of query results by last modified date, the service implementation of the WPSAPI must support the use of `oracle.webcenter.search.AttributePredicate`, accessible from the query object, that uses `oracle.webcenter.search.AttributeConstants.DCMI_MODIFIED`. The supported comparators must include the following:

- `oracle.webcenter.search.ComparatorConstants.GREATER_THAN`
- `oracle.webcenter.search.ComparatorConstants.GREATER_THAN_OR_EQUALS`

They may also include the following:

- `oracle.webcenter.search.ComparatorConstants.EQUALS`
- `oracle.webcenter.search.ComparatorConstants.NOT_EQUALS`
- `oracle.webcenter.search.ComparatorConstants.LESS_THAN`
- `oracle.webcenter.search.ComparatorConstants.LESS_THAN_OR_EQUALS`

The literals for comparison should be of type `java.util.Calendar`.

45.3.9.3.4 Complex Date Condition You can use the `BETWEEN` clause for dates. The complex date condition, instead of being a simple `AttributePredicate`, is a `ComplexPredicate` that contains one `GREATER_THAN AttributePredicate` and one `LESS_THAN AttributePredicate` with an `AND` conjunction operator. This `ComplexPredicate` must apply the `AND` operator with the rest of the fields (that is, keywords and person).

To differentiate between this case and the default case, service authors must check on the type of the predicate that is meant for the date condition. If it is an `AttributePredicate`, then it is the default case. If it is a `ComplexPredicate`, then it is a special case.

45.3.9.3.5 Person Condition For Search to support filtering of query results by person, the service implementation of the WPSAPI must support `oracle.webcenter.search.AttributePredicate`, accessible from the query object, that uses `oracle.webcenter.search.AttributeConstants.WPSAPI_MODIFIER`.

The comparators supported must include the following:

- `oracle.webcenter.search.ComparatorConstants.EQUALS`
- `oracle.webcenter.search.ComparatorConstants.CONTAINS`
- `oracle.webcenter.search.ComparatorConstants.STARTS_WITH`
- `oracle.webcenter.search.ComparatorConstants.ENDS_WITH`

The literals for comparison should be of type `java.lang.String`.

45.3.9.3.6 Complex Person Condition Service authors must check on the type of predicate that is meant for the person condition. An `AttributePredicate` is the default case.

45.3.9.3.7 Access The Search service (as a caller) expects that the query's `getPredicate()` method likely returns an `oracle.webcenter.search.ComplexPredicate` that joins an `oracle.webcenter.search.TextPredicate` holding the keyword, and an `oracle.webcenter.search.ComplexPredicate` holding the date condition, and an `oracle.webcenter.search.AttributePredicate` holding the person condition.

45.3.9.3.8 Selectable Attributes The following selectable attributes/columns should be supported by your search adapter in its search implementation:

- **Title** (Required)

  For each resource returned, the Search service user interface can display the title column, if provided. It is strongly recommended that the title column comes with the QName `AttributeConstants.DCMI_TITLE`.

- **Resource ID** (Required)

  For resources to communicate a unique identifier that allows the Search service to render a link to navigate to the resource view (or any declared views) of the service, the column with the name `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` must be returned as a column accessible from a row within the `QueryResult<Row>`.

  The Search service extracts the `DCMI_IDENTIFIER` column and feeds it into the resource viewer as the value of the `resourceId` parameter when the link is clicked.

  If your resource view (or any declared views) requires multiple columns in the primary key, then try to bundle all of them into a composite string value for the `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` column. The Search service treats that opaquely and passes it to your resource view, within which it may deconstruct the string into the various parts of the primary key.

- **Resource Type**

  For each resource returned, the Search service user interface can display the type column, if provided. The type column comes with the QName `AttributeConstants.DCMI_TYPE`. If the `DCMI_TYPE` column is not found, then the value of the `DCMI_FORMAT` column is used in its place. In your resource viewer, you can make use of the resource type to aid in the rendering of a resource.

- **Resource URL** (in place of Resource ID)

  In the absence of an `oracle.webcenter.search.AttributeConstants.DCMI_IDENTIFIER` column, WebCenter Portal's Search service constructs the link from the value of the column specified by `oracle.webcenter.search.AttributeConstants.DCMI_URI`. With absolute URLs, the `DCMI_URI` can launch a link from the search results to the URL.

  This allows for services, such as the Documents service and the Page service, to invoke the browser plug-in to render various results link targets, instead of invoking a resource view.

- **Last Modified Date**

For each resource returned, the Search service user interface can display the last modified date column, if provided. To mirror the searchable attribute of the last modified date, it is important that users can see the last modified date as a returned column in the search result.

For most users, the last modified date is more relevant than the creation date. To have better consistency between the searchable attributes and the selectable attributes (so that users know what date to search with after seeing some values in the columns), it is strongly recommended that service authors return `AttributeConstants.DCMI_MODIFIED` rather than `AttributeConstants.DCMI_CREATED`. The last modified date is of type `java.util.Calendar`.

- **Creator**

  For each resource returned, the Search service user interface can display the creator and last modified by column, if provided. The search user interface combines them in the same column as a list of contributors for the found resource. The QNames to use are `AttributeConstants.DCMI_CREATOR` and `AttributeConstants.WPSAPI_MODIFIED`, respectively. The creator is of type `java.lang.String`.

- **Icon**

  For each resource returned, the Search service user interface can display the icon column, if provided. The icon column comes with the QName `AttributeConstants.WPSAPI_ICON_PATH` and corresponds to a valid path to an icon file that is accessible in the class path. The icon path is of type `java.lang.String`.

- **Size**

  For each resource returned, the Search service user interface can display the size column, if provided. The size column comes with the QName `AttributeConstants.DCMI_EXTENT` and corresponds to the size of the resource found. The size is of type `java.lang.Number`.

45.3.9.4 What Happens at Runtime

When you perform a search from the main view or toolbar, it should seamlessly include your new search source. For testing purposes, be sure to enter criteria that yields a result in your newly-added search source.

WebCenter Portal's Search service invokes a resource viewer using the Resource Action Handling framework. For information about registering a resource viewer to allow custom components to be found using Search, see Section 7.3, "Extending Your Framework Application with Custom Components."

45.3.10 Customizing the Search UI Without WebCenter Portal's Search Service APIs

You can develop a custom search user interface in a Framework application without WebCenter Portal's Search service API, but still with Oracle SES as the application's search engine by using the Oracle SES Web Service Query API to do search functions. Also, action handling for search results can be managed with WebCenter Portal's Resource Action Handling framework, so that when an application resource (for example, a document) is clicked in the search results, the resource is shown in the viewer of the resource (for example, the Documents resource viewer). Moreover, if the application uses WebCenter Portal Navigation, then certain resource types can be shown in the navigation node of the resource.

This section describes the required steps to enable Resource Action Handling for handling search result action.

1. Use the Document Service Manager provided by WebCenter Portal to crawl documents.

   See Also: Section "Setting Up Oracle SES to Search Documents" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*

   The Document Services Manager creates an attribute called `wc_url` in the Oracle SES index for each document. The value of the `wc_url` attribute is a URL that goes to the Resource Action Handling module of the application. The URL uses the value of the `WebCenter URL Prefix` parameter in the Document Service instance as the prefix.

   Set the parameter value as the host and port where the application is deployed, plus the context root; for example:

   ```
   http://myhost:8888/DocumentsServer
   ```

2. Use the `wc_url` attribute for action handling in the search UI.

   Typically, the search UI uses the URL attribute for action handling. To use Resource Action Handling for action handling of search results of the Documents resource type, use the `wc_url` attribute. Note that only search results with the Documents resource type have the `wc_url` attribute. Therefore, this must be done conditionally in the search UI code.

3. Specify WebCenter Portal's Resource Action Handling class for click action of search result.

   When a search result is clicked, the resource is rendered in the resource viewer of the resource, if available. The current search results page is over-written. Use Resource Action Framework to change this behavior.

   Update the `adf-config.xml` file of the application to specify the Resource Action Handling class to use.

   Example 45–10 shows how to use `PopUpResourceActionViewHandler` to show the resource in a popup window when a search result is clicked.

Example 45–10 Specifying PopUp Behavior Handler

```
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
  <wpsC:adf-service-config
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
    <resource-handler
class="oracle.webcenter.framework.resource.view.PopUpResourceActionHandler"/>
  </wpsC:adf-service-config>
```

   Example 45–11 shows how to use `NavigationResourceActionHandler` to show the resource in the navigation node of the resource when a search result is clicked.

Example 45–11 Specifying the Navigation Handler

```
xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
  <wpsC:adf-service-config
xmlns="http://xmlns.oracle.com/webcenter/framework/service">
    <resource-handler
```

```
class="oracle.webcenter.portalframework.sitestructure.handler.NavigationResourceAc
tionHandler"/>
  </wpsC:adf-service-config>
```

45.3.11 Troubleshooting the Search Service

This section describes common problems and solutions for the Search service.

Problem

There is a set timeout for querying each service. If a particular service does not return search results in the allotted time, then it times out. The Announcements and Discussions services are susceptible to timeouts. (In WebCenter Portal: Spaces, pages and spaces could have timeouts from search execution.)

Solution

To improve performance at design time, you can increase the values (in milliseconds) of `timeoutMs` and prepare `TimeoutMs` in `adf-config.xml`. The following is the relevant fragment of `adf-config.xml`, found in the Application Resources panel in the `ADF META-INF` folder:

```
<execution-properties timeoutMs="3000"prepare TimeoutMs="1000"/>
```

At runtime, you can configure timeouts using WLST or Fusion Middleware Control.

46

Integrating the Activity Graph Service

This chapter explains how to integrate the Activity Graph service in a WebCenter Portal: Framework application at design time.

This chapter includes the following sections:

- Section 46.1, "Introduction to the Activity Graph Service"
- Section 46.2, "Basic Configuration for the Activity Graph Service"
- Section 46.3, "Advanced Information for the Activity Graph Service"
- Section 46.4, "Extending the Activity Graph Service"
- Section 46.5, "Troubleshooting the Activity Graph Service"

For more information about managing and using the Activity Graph service, see:

- "Managing the Activity Graph Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- "Working with the Activity Graph Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

46.1 Introduction to the Activity Graph Service

The advent of Enterprise 2.0 has enabled users to contribute content at unprecedented rates. In view of this, there is now a need for better information retrieval technologies that are seamlessly integrated with Enterprise 2.0 products, such as Framework applications. The Activity Graph service provides the solution by leveraging collective intelligence to benefit search and social applications.

This section provides an overview of the Activity Graph service, its task flows, its benefits, and the underlying architecture. It includes the following subsections:

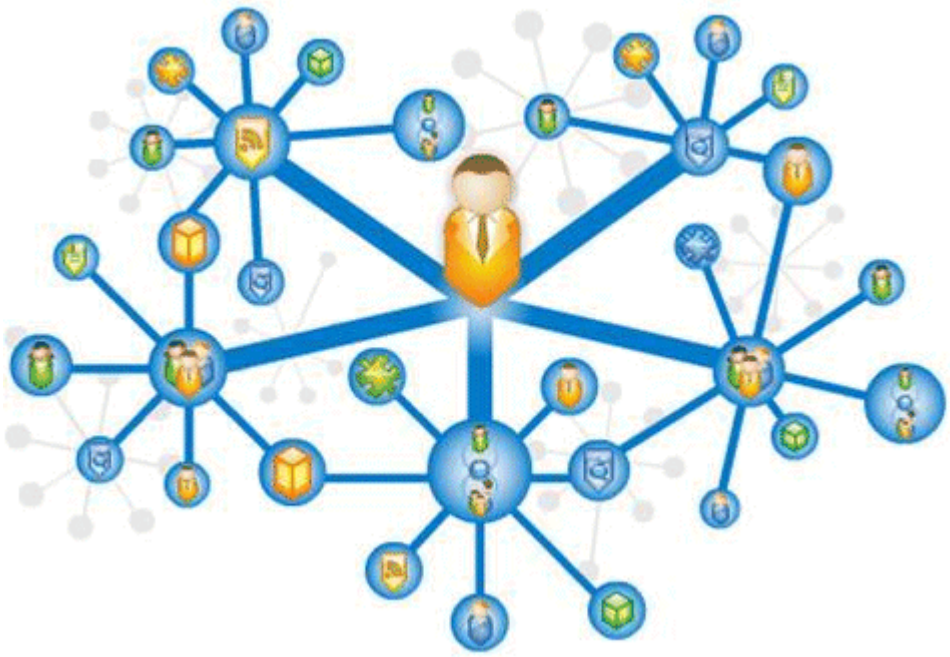
- Section 46.1.1, "Understanding the Activity Graph Service"
- Section 46.1.2, "Requirements for the Activity Graph Service"
- Section 46.1.3, "What Happens at Runtime"

46.1.1 Understanding the Activity Graph Service

The Activity Graph service provides suggestions of people that a user may be interested in connecting with, based on existing connections and shared interaction with objects within the application. It also directs users to spaces or content that may be of interest, based on similar interactions with those spaces or items that the user is currently viewing.

The Activity Graph service presents these suggestions based on data gathered and analyzed by the Activity Graph engine. The Activity Graph engine provides a central repository for actions that are collected by enterprise applications. Thinking in terms of a mathematical graph, application users and the enterprise content with which they interact are *nodes*, and the actions between users and between users and content are *directed edges* (Figure 46–1).

Figure 46–1 An Activity Graph



In an enterprise, this analysis of the interaction of people with other people and with content produces similarity scores for making contextual recommendations based on an extensible set of actions, such as viewing, editing, tagging, and so on. For online vendors, for example, such contextual recommendations help provide a selection of suggested additional purchases. In a social networking environment, recommendations suggest additional connections to make based on the friends of the friends you have already connected with.

The Activity Graph engine also calculates an activity rank for content and passes this information to Oracle Secure Enterprise Search to enable more relevant content to appear higher in search results. For more information see the section "Setting Up Activity Rank for Oracle Secure Enterprise Search" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Node Classes

Activity graph nodes are grouped into classes. The default *node classes* for WebCenter Portal are:

- Users (`WC.user`)
- Spaces (`WC.group-space`)
- Documents (`WC.document`)
- Wiki Pages (`WC.wiki-page`)

- Blogs (`WC.blog`)
- Discussion Topics (`WC.topic`)

If you want to extend the Activity Graph engine to integrate with other applications, then you can create custom node classes for the objects in those applications. For more information, see Section 46.4.1, "Defining Custom Node Classes."

Actions

An *action* is a specific type of event. It has a source and a target. For example, if a user, say Monty, looks at a document, that is a `view` action with Monty as the source and the document as the target. Other actions between users and items include `create`, `like`, and `tag`. Actions can also occur between two users, for example, when two users `connect` with each other.

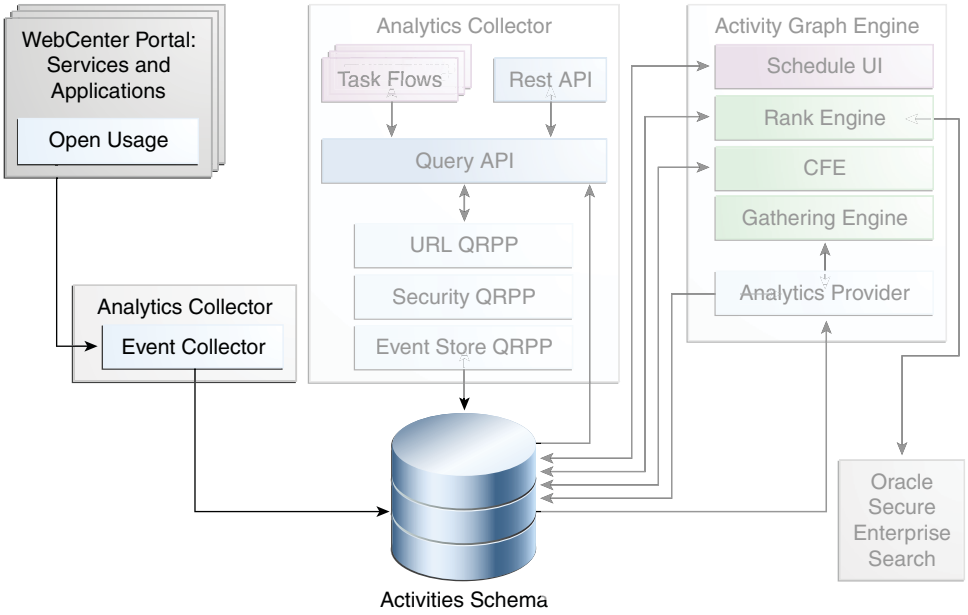
Table 46–1 lists the default actions defined for WebCenter Portal.

Table 46–1 WebCenter Portal Default Actions

Action URN	Description
`connect`	Connecting with another user.
`create`	Creating a space, document, wiki page, blog, topic, or message.
`edit`	For all items except spaces: ■ Checking in a new version of document ■ Editing a wiki page or blog ■ Replying to a topic ■ Editing a message For spaces: ■ All of the above plus ■ Creating, editing, or deleting a page ■ Creating, editing, or deleting a list ■ Creating, editing, or deleting an event
`comment`	Commenting on a document, wiki, or blog
`like`	Liking a document, wiki, blog, or message
`tag`	Tagging a document, wiki, or blog
`view`	Viewing a space page, document, wiki page, blog entry, or discussion topic
`download`	Downloading a document, wiki, or blog
`edit-count`	Checking in a new version of a document, editing a wiki page or blog, replying to a topic, editing a message
`view-count`	Viewing a document, wiki page, blog, or discussion topic

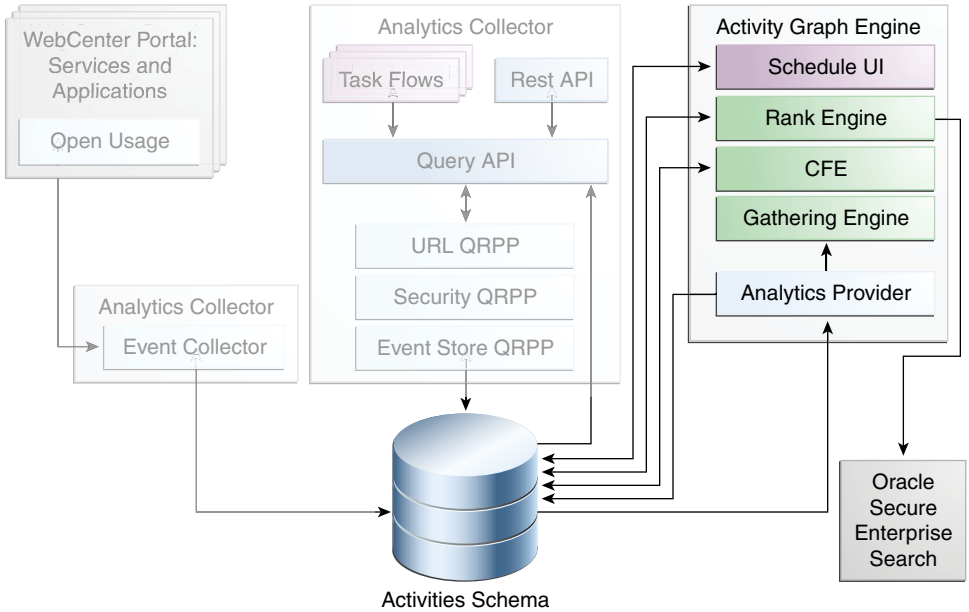
When one of these WebCenter Portal actions occurs, for example, Monty `view`ing his document, it is picked up by the Events Collector, a component of WebCenter Portal's Analytics, and placed in an event table in the Activities database (Figure 46–2).

Figure 46–2 Data Collection by WebCenter Portal's Analytics



When the activity data gathering process starts, Analytics Activity Provider reads actions from the Analytics event tables and uses a registered set of mappings to generate activities (Figure 46–3). An *activity* is one occurrence of an action and is used to determine *relations*, aggregated occurrences of actions, which are stored in the relation tables. For example, the fact that Monty has viewed this particular document five times is a relation. Information in the relation tables is used to determine recommendations and search ranks.

Figure 46–3 Data Gathering by the Activity Graph Engine



If you want to integrate the Activity Graph engine with other applications, then you can create custom actions for the actions that users perform in those applications. For more information, see Section 46.4.2, "Defining Custom Actions."

Similarity Calculations

The Activity Graph Query API is a Java API, used by the Activity Graph service task flows, that queries the relation tables for recommendations using a recipe (Figure 46–4). A *recipe* is a weighted list of similarity calculations. A *similarity calculation* provides a similarity score (a number between zero and one) that designates how similar two objects are to each other given a specific criterion. The weighting of each calculation determines its significance in deciding the overall recommendation score. Recommendations are ordered by their total recommendation score.

Figure 46–4 Recommendations Calculation by the Activity Graph Service

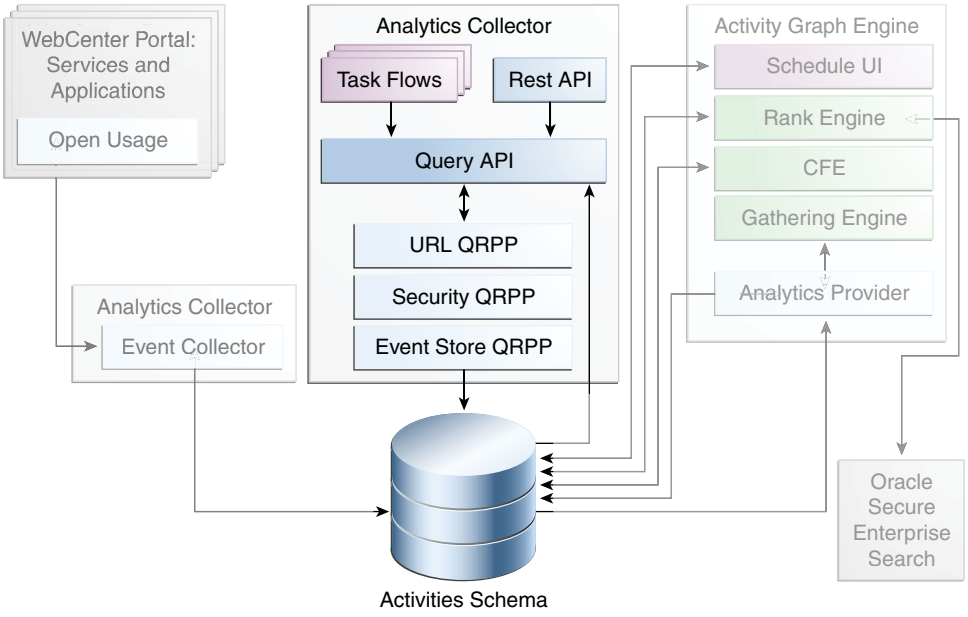


Table 46–2 lists the similarity calculations used by the Activity Graph service task flows (for information about the task flows, see Section 46.2.3.1, "Activity Graph Service Task Flows").

Table 46–2 Activity Graph Service Similarity Calculations

URN	Applies to	Description	Used by Task Flow
`item-edit`	Documents Wiki pages Blogs Topics	Users who edited the current item also edited the recommended item	Similar Items
`item-like`	Documents Wiki pages Blogs Topics	Users who liked the current item also liked the recommended item	Similar Items
`item-comment`	Documents Wiki pages Blogs Topics	Users who commented on the current item also commented on the recommended item	Similar Items

Table 46–2 (Cont.) Activity Graph Service Similarity Calculations

URN	Applies to	Description	Used by Task Flow
`item-tag`	Documents Wiki pages Blogs Topics	Users who tagged the current item also tagged the recommended item	
`item-all`	Documents Wiki pages Blogs Topics	Users who interacted in any way with the current item, including viewing it, also interacted in some way with the recommended item	Similar Items
`gs-edit`	Spaces	Users who participated in the current space also participated in the recommended space	Similar Spaces
`gs-all`	Spaces	Users who interacted in any way with the current space, including viewing pages or content in the space, also interacted in some way with the recommended space	Similar Spaces
`user-connect`	Users	The current user shares a number of connections with the recommended user	Recommended Connections
`user-edit`	Users	The current user edited some items that the recommended user also edited	Recommended Connections
`user-like`	Users	The current user likes some items that the recommended user also likes	Recommended Connections
`user-comment`	Users	The current user commented some items that the recommended user also commented on	Recommended Connections
`user-tag`	Users	The current user tagged some items that the recommended user also tagged	Recommended Connections
`user-all`	Users	The current user interacted in some way, including viewing, with some items that the recommended user also interacted with	Recommended Connections

You can edit these similarity calculations to change the weightings. If you want to integrate the Activity Graph engine with other applications, then you can create additional similarity calculations. For more information, see Section 46.4.3, "Defining Custom Similarity Calculations.".

Query Result Post-Processors (QRPPs)

After the initial list of recommendations for a particular object is generated, the results can be filtered into something more appropriate and useful to present to users. This is achieved using Query Result Post-Processors (QRPPs). QRPPs take the current list of recommendation results return a modified list as output. A QRPP may filter out recommendations, for example by removing recommendations for objects that the current user is not permitted to see, or may add or modify result metadata.

WebCenter Portal provides three QRPPs (run in the following order):

- **WebCenter Portal Security QRPP**—Filters out recommendations for objects for which the current user does not have view permissions by calling the Resource Authorizer, which is the same WebCenter Portal component used by the Search service for view authorization.
- **Analytics Metadata QRPP**—Retrieves additional metadata, for example, description and icon URL, for each recommended object from the Analytics tables.
- **WebCenter Portal URL QRPP**—Retrieves a URL for each recommended object by calling the Resource Action Handler, which is the same WebCenter Portal component used by the Search service.

You can create your own QRPPs to further filter the results or to integrate the Activity Graph engine with other applications. For more information, see Section 46.4.6, "Registering Custom QRPPs."

Activity Graph Service Task Flows

Recommendations are presented to users via the Activity Graph service task flows. The Activity Graph service provides the following task flows:

- **Recommended Connections**—Recommends users that the current user may want to connect with based on his or her current connections and shared interactions with objects in the application.
- **Similar Spaces**—Suggests spaces with which users have interacted in a similar way as the current space.
- **Similar Items**—Suggests items with which users have interacted in a similar way as the current item.

For more information, see Section 46.2.3.1, "Activity Graph Service Task Flows."

46.1.2 Requirements for the Activity Graph Service

The Activity Graph service requires that the Activity Graph engine has been installed and configured. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

In addition, in your application you must create a connection to WebCenter Portal's schema and to the Activities database. For more information, see Section 7.2.2, "Setting Up a Database Connection."

The application must be configured to send usage events to the Analytics Event Collector. For more information, see the section "Registering an Analytics Collector for

Your Application" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Before the Activity Graph service can make recommendations, the Activity Graph engines must have been run at least once to gather the data and calculate similarity scores. For more information see the section "Preparing Data for the Activity Graph Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The items suggested in the Similar Items task flow depend on the services that are available in your application. For example, documents are only recommended if the Documents service is available. For information about making a service available in your application, refer to the appropriate chapter for that service. An item can also be filtered out of the recommendations by the Resource Authorizer of the service that owns the item.

46.1.3 What Happens at Runtime

At runtime, the context is provided for the Activity Graph service task flow:

- For the Recommended Connections task flow, the context is the current user.
- For the Similar Spaces task flow, the context is the current space.
- For the Similar Items task flow, the context is a WebCenter Portal resource, provided through task flow input parameters or through ADF UI events.

The task flow then queries the Activity Graph database for recommendations through the Activity Graph Query API, using the recipe provided in the task flow parameters. The list of recommendations returned from the Query API may be filtered by QRPPs before being listed in the task flow.

For more information about the service at runtime, see the section "Working with the Activity Graph Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

46.2 Basic Configuration for the Activity Graph Service

This section describes required steps for adding the Activity Graph service to your application. It includes the following subsections:

- Section 46.2.1, "Configuration Roadmap for the Activity Graph Service"
- Section 46.2.2, "Setting Up Connections for the Activity Graph Service"
- Section 46.2.3, "Adding the Activity Graph Service at Design Time"

46.2.1 Configuration Roadmap for the Activity Graph Service

The flow chart (Figure 46–5) and table (Table 46–3) in this section provide an overview of the prerequisites and tasks required to get the Activity Graph service working in Framework applications.

Figure 46–5 Configuring the Activity Graph Service for WebCenter Portal: Framework Applications

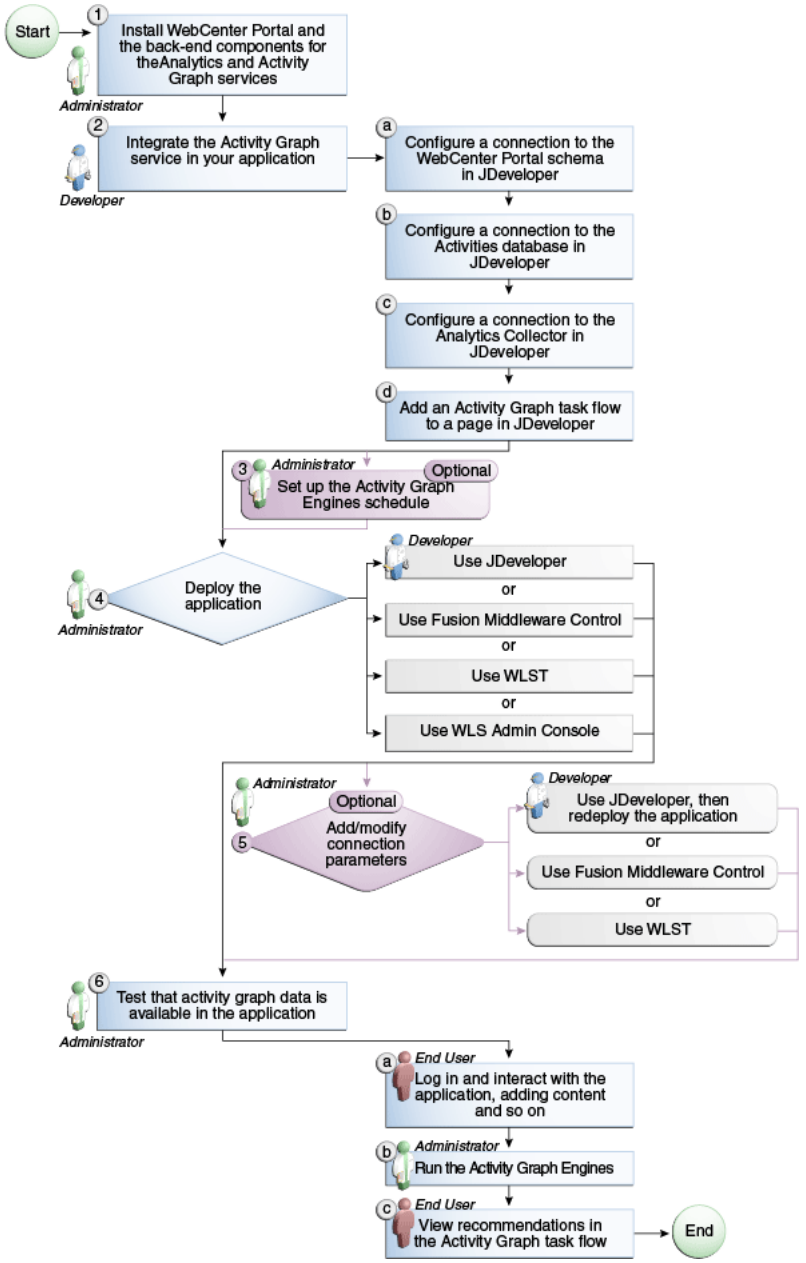


Table 46–3 Configuring the Activity Graph Service for WebCenter Portal: Framework Applications

Actor	Task	Sub-task
Administrator	**1.** Install WebCenter Portal and the back-end components for the Analytics and Activity Graph services	
Developer	**2.** Integrate the Activity Graph service in your application	**2.a** Configure a connection to WebCenter Portal's schema in JDeveloper **2.b** Configure a connection to the Activities database in JDeveloper **2.c** Configure a connection to the Analytics Collector in JDeveloper **2.d** Add an Activity Graph task flow to a page in JDeveloper
Administrator	**3.** (Optional) Set up the Activity Graph Engines schedule	
Developer/ Administrator	**4.** Deploy the application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Console (Administrator)	
	5. (Optional) Add/modify connection parameters using one of the following tools: ■ JDeveloper, then redeploy the application (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator)	
End User/ Administrator	**6.** Test that activity graph data is available in the application	**6.a** Log in and interact with the application, for example, by adding content (End User) **6.b** Run the Activity Graph Engines (Administrator) **6.c** View recommendations in the Activity Graph task flow, for example, the Recommended Connections task flow on your Profile page (End User)

46.2.2 Setting Up Connections for the Activity Graph Service

The Activity Graph service requires a the following connections:

- **WebCenter Portal schema**—This connection is used by the WebCenter Portal Security and WebCenter Portal URL QRPPs when querying the WebCenter Portal Resource Authorizer and Resource Action Handler respectively. For information about how to create this connection, see Section 7.2.2, "Setting Up a Database Connection."
- **Activities schema**—This connection is used to query recommendations and obtain metadata for results. This is the same connection as used by the Analytics Event Collector and the Activity Graph engine. If you have an existing connection, then that connection is used. If you do not have an existing connection, then you must

create one. For information about how to create this connection, see Section 7.2.2, "Setting Up a Database Connection."

- **Analytics Event Collector**—This connection is used to send usage events to the Analytics Event Collector. For more information, see the section "Registering an Analytics Collector for Your Application" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

> **Note:** While you can set up the connections to back-end servers at design time in JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.2.3 Adding the Activity Graph Service at Design Time

This section provides descriptions of the Activity Graph service task flows and steps you through the addition of the Activity Graph service to your application. It includes the following subsections:

- Section 46.2.3.1, "Activity Graph Service Task Flows"
- Section 46.2.3.2, "How to Add Activity Graph Service Task Flows to a Page"
- Section 46.2.3.3, "How to Modify Activity Graph Service Task Flow Parameters"
- Section 46.2.3.4, "Activity Graph Service Task Flows and Task Flow Parameters"

46.2.3.1 Activity Graph Service Task Flows

This section describes and illustrates the task flows available through the Activity Graph service.

Table 46–4 lists the task flows provided by the Activity Graph service.

Table 46–4 Activity Graph Service Task Flows

Task Flow	Description
Recommended Connections	This task flow enables viewing and connecting to users whom the Activity Graph engine has calculated to be similar to the current user. For more information, see Section 46.2.3.1.1, "The Recommended Connections Task Flow."
Similar Spaces	This task flow enables viewing and interacting with spaces that the Activity Graph engine has calculated to be potentially of interest to the current user. For more information, see Section 46.2.3.1.2, "The Similar Spaces Task Flow."
Similar Items	This task flow enables viewing and interacting with the WebCenter Portal items that the Activity Graph engine has calculated to be potentially of interest to the current user. For more information, see Section 46.2.3.1.3, "The Similar Items Task Flow."

46.2.3.1.1 The Recommended Connections Task Flow

The Recommended Connections task flow enables viewing and connecting to users who have been calculated to be similar to the current user.

The Recommended Connections task flow is available by default on each user's Profile page.

For information about how to use this task flow at runtime, see the section "Working with the Recommended Connections Task Flow" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Table 46–5 shows the similarity calculations and their weightings used in the default recipe for the Recommended Connections task flow. You can change this recipe by editing the `similarityURNList` task flow binding parameter. For more information, see Section 46.2.3.3, "How to Modify Activity Graph Service Task Flow Parameters."

Table 46–5 Default Recipe for Recommended Connections

Similarity Calculation	Weight
user-connect	100
user-edit	50
user-like	50
user-comment	10
user-tag	10
user-all	1

The user with the highest overall recommendation score is displayed first in the task flow. For each recommended user, if the highest scoring related similarity calculation has an associated reason string, that string is displayed underneath the user's name and description to provide details for why the user was recommended.

If no reason string is defined for the top similarity function, nothing is displayed. You can edit the similarity calculation reason strings, or provide additional strings. For more information, see the section "Customizing Reason Strings for Similarity Calculations" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.2.3.1.2 The Similar Spaces Task Flow

The Similar Spaces task flow enables viewing and interacting with spaces that have been calculated to be potentially of interest to users of the current space. The suggested spaces are not specific to the current user, however the task flow lists only those spaces that the current user has permission to see.

For information about how to use this task flow at runtime, see the section "Working with the Similar Spaces Task Flow" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Table 46–6 shows the similarity calculations and their weightings used in the default recipe for the Similar Spaces task flow. You can change this recipe by editing the `similarityURNList` task flow binding parameter. For more information, see Section 46.2.3.3, "How to Modify Activity Graph Service Task Flow Parameters."

Table 46–6 Default Recipe for Similar Spaces

Similarity Calculation	Weight
gs-edit	10
gs-all	1

The space with the highest overall recommendation score is displayed first in the task flow. For each recommended space, if the highest scoring related similarity calculation has an associated reason string, that string is displayed underneath the space's name and description.

If no reason string is defined for the top similarity calculation, nothing is displayed. You can edit the similarity calculation reason strings, or provide additional strings. For more information, see the section "Customizing Reason Strings for Similarity Calculations" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.2.3.1.3 The Similar Items Task Flow The Similar Items task flow enables viewing and interacting with the WebCenter Portal items that the Activity Graph engine has calculated to be potentially of interest to users of the currently selected item. The suggested items are not specific to the current user, however the task flow lists only those items that the current user has permission to see.

Items of interest may include a wiki page, a blog post, a document, or a discussion topic.

The Similar Items task flow determines the context for its recommendations based on items selected on other task flows on the page. This information is provided through task flow input parameters (typically using an EL expression) or through the `WebCenterResourceSelected` ADF UI event. The task flows that determine the context of the Similar Items task flow are:

- Documents service task flows:
  - Recent Documents
  - Document List Viewer
  - Document Navigator
  - Document Explorer
  - Folder Viewer
  - Document Manager
  - Document Browser
- Discussions Manager task flow

The Similar Items task flow is included in the Document Explorer Related Items pane for files to show items similar to the currently viewed file. For more information, see the section "Viewing and Adding Tags, Links, and Recommendations to a File" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

For information about how to use this task flow at runtime, see the section "Working with the Similar Items Task Flow" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Table 46–7 shows the similarity calculations and their weightings used in the default recipe for the Similar Items task flow. You can change this recipe by editing the `similarityURNList` task flow binding parameter. For more information, see Section 46.2.3.3, "How to Modify Activity Graph Service Task Flow Parameters."

Table 46–7 Default Recipe for Similar Items

Similarity Calculation	Weight
item-edit	100

Table 46–7 (Cont.) Default Recipe for Similar Items

Similarity Calculation	Weight
item-like	50
item-comment	20
item-tag	10
item-all	1

The item with the highest overall recommendation score is displayed first in the task flow. For each recommended item, if the highest scoring related similarity calculation has an associated reason string, that string is displayed underneath the item's name and description to provide details for why the item was recommended.

If no reason string is defined for the top similarity function, nothing is displayed. You can edit the similarity calculation reason strings, or provide additional strings. For more information, see the section "Customizing Reason Strings for Similarity Calculations" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.2.3.2 How to Add Activity Graph Service Task Flows to a Page

The section describes how to add the Recommended Connections task flow to an application page. The steps provided here are the same for all Activity Graph service task flows.

To add the Recommended Connections task flow to your Framework application:

1. Prepare your application as described in Section 46.1.2, "Requirements for the Activity Graph Service."
2. Open the page on which you want to add the task flow.
3. In the Resource Palette, open **My Catalogs**, then **WebCenter Portal - Services Catalog**, then the **Task Flows** folder.
4. Drag and drop the Recommended Connections task flow (or whichever Activity Graph service task flow you prefer) onto the JSF (`.jspx`) page.
5. Select **Region** from the resulting context menu.

   You may be prompted to add the Activity Graph service ADF library (`activitygraph-service-view.jar`) to the project. Confirm by clicking **Add Library**. This operation may take a moment to complete.
6. In the Edit Task Flow Binding dialog, enter values for the parameters, or accept the defaults.

   For more information, see Section 46.2.3.4, "Activity Graph Service Task Flows and Task Flow Parameters."
7. Click **OK**.

   The task flow is added to the page, and the project's libraries are configured to run the task flow.
8. Save and run your page.

46.2.3.3 How to Modify Activity Graph Service Task Flow Parameters

Each Activity Graph service task flow has a set of required and optional task flow binding parameters. These provide a means of capturing information that is useful to the task flow's successful function.

In addition to providing required values for successful task flow rendering, you can use task flow binding parameters to customize the appearance and behavior of a task flow instance. For example, you can use parameter values to determine whether headers and footers are rendered, the number of rows and columns of information to show, whether to apply a filter to returned data, and the like.

You can provide task flow binding parameter values when you drag and drop a task flow onto an application page. Doing so opens the Task Flow Bindings dialog (for more information, see Section 46.2.3.2, "How to Add Activity Graph Service Task Flows to a Page").

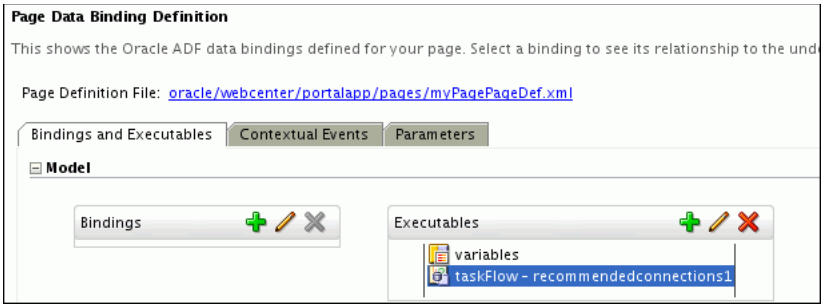
You can also adjust task flow binding parameter values after you have placed a task flow on a page.

To access the Edit Task Flow Binding dialog:

1. Click the **Bindings** tab at the bottom of the page (next to the **Source** tab) to go to the Bindings view.
2. Under **Executables**, you'll see the task flow you added.

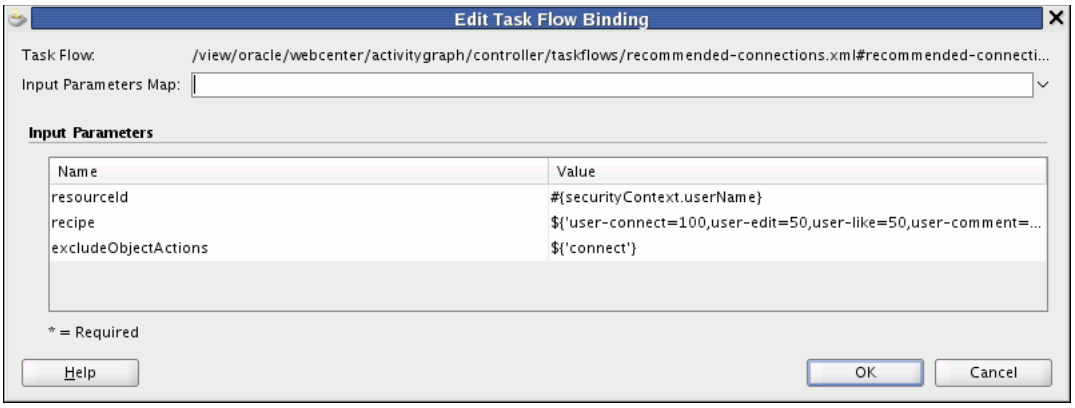
   Figure 46–6 shows an example of a Recommended Connections task flow in the Executables section.

Figure 46–6 Recommended Connections in the Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog (Figure 46–7), revise the binding parameter values as required (for more information, see Section 46.2.3.4, "Activity Graph Service Task Flows and Task Flow Parameters").

Figure 46–7 Edit Task Flow Binding Dialog for Recommended Connections



5. When you are finished, click **OK.**
6. Save and run your page to see the results.

46.2.3.4 Activity Graph Service Task Flows and Task Flow Parameters

Table 46–8 lists and describes task flow binding parameters applicable to the Activity Graph service.

Table 46–8 Activity Graph Service Task Flow Binding Parameters

Parameter	Task Flows	Description
`resourceId`	Recommended Connections Similar Spaces Similar Items	The ID of the person, space, or item to use as a context for recommendations. For connections, the default value is the expression language token for the currently logged in user. For spaces, the default value is the expression language token for the resource ID of the space containing the task flow. For items, the value is derived from the selection event.
`classURN`	Similar Items	The node class of the context. By default the information is derived from the selection event.
`objectName`	Similar Spaces Similar Items	The name of the space or item to use as a context for recommendations. For spaces, the default value is the expression language token for the resource ID of the space containing the task flow. For items, the value is derived from the selection event.

Table 46–8 (Cont.) Activity Graph Service Task Flow Binding Parameters

Parameter	Task Flows	Description
`recipe`	Recommended Connections Similar Spaces Similar Items	A comma separated list of registered similarity calculation = weight pairs. The list determines what recommendations appear and how they are ordered. The default recipes for each task flow are listed in Section 46.2.3.1, "Activity Graph Service Task Flows."
`classURNRestrictions`	Similar Items	A comma separated list of the node classes of objects to exclude from the recommendations. By default this is empty, meaning that there are no restrictions on which classes of objects may be returned.
`excludeObjectActions`	Recommended Connections Similar Spaces Similar Items	A comma separated list of registered actions. The task flow will not show people, spaces, or items on which the logged in user took any of the listed actions. For example, the default value for the Similar Items task flow is create, edit, comment, tag, meaning that users are not recommended items which they themselves created, edited, commented on, or tagged. The WebCenter Portal default actions are listed in Table 46–1.
`serviceId`	Similar Items	Used in conjunction with `resourceId`, and optionally `resourceType`, as an alternative to `classURN` for identifying the node class of an object.
`resourceType`	Similar Items	Used in conjunction with `resourceId` and `serviceId`, as an alternative to `classURN` for identifying the node class of a WebCenter Portal object. For example `resourceType` is used by the Documents service to identify whether an object is a wiki, blog, or document.
`suppressPopup`	Similar Items	Determines whether links in the task flow launch content in inline popups (`false`) or in browser windows (`true`). The default value is `false`.

46.3 Advanced Information for the Activity Graph Service

This section describes advanced features available with the Activity Graph service. It includes the following subsections:

- Section 46.3.1, "Using the Activity Graph Data Control"

- Section 46.3.2, "Using the Activity Graph Service REST APIs"

46.3.1 Using the Activity Graph Data Control

The Activity Graph service provides a data control that enables you to create your own user interface (task flows) for the service.

The Activity Graph data control exposes two methods to query Activity Graph for suggestions of similar items and similar users, and one method to record items the user is not interested in.

- `getSimilarObjects`—Returns a list of `SuggestionWrappers` for a given source and similarity calculation, with choices for complex similarity calculation recipes and filtering of node classes and actions.
- `recordNotInterested`—Records that the user is not interested in a given object.

> **Note:** For convenience, the `getSimilarObjects` method takes string arguments instead of the list and map arguments used at the lower levels of the APIs, since that is typically how you would configure them as task flow input parameters.

Typically, there are three things that you need to bind your data control to a method: a `MethodAction`, a `MethodIterator` to iterate through the results of the method, and a `Tree` to pick attributes of the resulting data structure. You can create these manually in a page definition file, or more easily, use the wizards that JDeveloper provides.

To add the Activity Graph data control to your project:

1. Add the **RecommendationDataControl** to your project (either by selecting it in the catalog, right-clicking and choosing **Add to Project** or by dragging and dropping it onto a JSP page (`.jspx`) in your project).
2. Drag an ADF Table to a JSP page.
3. When prompted, check the **bind to datasource** checkbox.
4. Navigate to the **RecommendationDataControl** and select the SuggestionWrapper result from the API method you desire. This will automatically create a method action binding and an iterator to the resultant collection.
5. Select the nodes that you want to expose. This will automatically create a Tree to access values in the suggestion.
6. Configure the parameters for the method action, typically these will be expressions that you would have defined as pageFlowScope variables.
   - `classURN`—The URN of the node class of the source for the recommendation.
   - `objectURN`—The URN of the source for the recommendation.
   - `userCredentialClass`—if null, the WebCenter Portal user type is used.
   - `recipe`—a comma separated list of similarity calculations to be used and their associated weights. For example:
     - For a Similar Items task flow:

       ```
       item-edit=100,item-like=50,item-comment=20,item-tag=10,item-all=1
       ```

     - For a Recommended Connections task flow:

```
user-connect=100,user-edit=50,user-like=50,user-comment=10,user-tag=10,
user-all=1
```

– For a Similar Spaces task flow:

```
gs-edit=10,gs-all=1
```

- `classURNRestrictionList`—a comma separated list of classes to filter the results.
- `excludeObjectActionList`—a comma separated list of action URNs to use to filter the results.
- maxResults—the maximum number of recommendations to return

46.3.2 Using the Activity Graph Service REST APIs

WebCenter Portal provides REST APIs to support the Activity Graph service. Use the Activity Graph service REST APIs to create your own interface for providing recommendations for connections, spaces, and items.

This section describes the REST APIs associated with the Activity Graph service. It includes the following subsections:

- Section 46.3.2.1, "Activity Graph Entry Point"
- Section 46.3.2.2, "Activity Graph Resource Type Taxonomy"
- Section 46.3.2.3, "Security Considerations"
- Section 46.3.2.4, "Activity Graph Resource Types"

For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

46.3.2.1 Activity Graph Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. For the Activity Graph service there are two entry points: one for recommendations and one for items. To find the entry points for the Activity Graph service, find the link element with one of the following `resourceTypes`:

```
urn:oracle:webcenter:activitygraph:recommendations
urn:oracle:webcenter:activitygraph:items
```

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Activity Graph service.

For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

46.3.2.2 Activity Graph Resource Type Taxonomy

When the client has identified the entry point to use, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 46.3.2.2, "Activity Graph Resource Type Taxonomy."

The taxonomy for the Activity Graph service is:

```
urn:oracle:webcenter:activitygraph:recommendations
   urn:oracle:webcenter:activitygraph:recommendations:recommendation
```

```
urn:oracle:webcenter:activitygraph:items
   urn:oracle:webcenter:activitygraph:items:item
```

46.3.2.3 Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

46.3.2.4 Activity Graph Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- Section 46.3.2.4.1, "urn:oracle:webcenter:activitygraph:recommendations"
- Section 46.3.2.4.2, "urn:oracle:webcenter:activitygraph:recommendations:recommendation"
- Section 46.3.2.4.3, "urn:oracle:webcenter:activitygraph:items"
- Section 46.3.2.4.4, "urn:oracle:webcenter:activitygraph:items:item"

46.3.2.4.1 urn:oracle:webcenter:activitygraph:recommendations Use this resource to identify the URI to use to retrieve (`GET`) recommended connections, spaces, or items based on their similarity to the specified object. The response from a `GET` operation includes each object in the requested list, and each object includes links used to operate on that object.

Nodes in the activity graph are identified by a combination of node class URN and object URN.

For example, to identify the user, monty, you can specify:

- `classURN=WC.user`
- `objectURN=monty`

The nodes provided out of the box are all WebCenter Portal resources (users, documents, spaces, and so on) and so have WebCenter Portal service and resource IDs. The Activity Graph REST APIs provide another way for you to identify these out of the box nodes using the `serviceId` and `objectURN` (which contains the resource ID).

For example, to identify the user, monty, you can specify:

- `serviceId=oracle.webcenter.people`
- `objectURN=monty`

If a service's objects are further classified by resource type, for example, the Documents service, you must also specify the resource type.

For example, to identify a particular document, you can specify:

- `serviceId=oracle.webcenter.doclib`
- `resourceType=document`
- `objectURN=document1`

Any new node classes that are created (that is, non-native WebCenter Portal objects) should be identified using node class and object URNs.

Navigation Paths to recommendations

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceIndex
   recommendations
```

Supported Methods for recommendations

The following methods are supported by this resource:

- GET
  - **request - Parameters:** `startIndex`, `itemsPerPage`, `utoken`

    For information about these common parameters, see "Common Request Query Parameters".

    The following additional parameters are available:

    * `classURN`—the node class URN that identifies the type of the object for which you are requesting recommendations. For example `WC.user`, `WC.group-space`.
    * `objectURN`—the object URN that provides a unique identifier for the object for which you are requesting recommendations. For example `monty`, `1000`.
    * `recipe`—a semicolon-separated list of similarity URNs and optional associated weights (indicated by a colon) that is used to determine which objects to recommend. For example, `gs-edit:10;gs-all:1`
    * `classURNRestrictions`—a comma-separated list of types of objects, identified by node class URN, to exclude from the recommendations
    * `excludeObjectActions`—a comma-separated list of actions, identified by action URN, used to exclude objects from the recommendations if the current user has performed that action on the object. For example, if the client is retrieving recommended spaces, the `gs-edit` action could be specified to exclude spaces that the current user has edited (since the user already knows about those spaces).
    * `serviceId`—the WebCenter Portal service ID that identifies the type of object for which you are requesting recommendations (you can use this instead of `classURN` for out of the box objects).
    * `resourceType`—The WebCenter Portal resource type of the object for which you are requesting recommendations, used in combination with `serviceId` if necessary.
    * `userCredentialClassURN`—the node class URN of the user exercising the REST API. The default value is `WC.user`. If you are integrating the Activity Graph engine with another application, this may need to be a different node class.
  - **response - body:** 0 or more recommendations

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to from recommendations

Table 46–9 lists the resource types that the client can link to from this resource.

Table 46–9 Related Resource Types for recommendations

rel	resourceType
self	urn:oracle:webcenter:activitygraph:recommendations

46.3.2.4.2 urn:oracle:webcenter:activitygraph:recommendations:recommendation The recommendation response contains the recommended objects and the URIs for use in accessing those objects.

Navigation Paths to recommendation

This section shows how the client can navigate through the hypermedia to access the recommendation resource:

```
resourceIndex
   recommendations
      recommendation
```

Read-only Elements for recommendation

Table 46–10 lists the read-only elements for the recommendations resource.

Table 46–10 Read-only Elements for recommendation

Element	Type	Description
score	Float	The overall score of this recommendation relative to the other recommendations in the list. This is the weighted sum of the component scores associated with each of the similarity URNs that comprise the recipe and is a floating point number between 0 and 1.
item	urn:oracle:webcenter:activitygraph:items:item	The recommended user, item, or space.
componentScores	A list of `componentScore` elements	A list of the component scores associated with the different similarity URNs in the recipe for the recommendation. A component score may have a reason and a link which can be used to retrieve the common items that the user and the recommended object have interacted with.

46.3.2.4.3 urn:oracle:webcenter:activitygraph:items Use this resource to identify the URI to use to retrieve (`GET`) objects that the source object and recommended object have in common. You can use this to determine the reasons why a particular object was recommended. The response from a `GET` operation includes each item in this collection of items, and each item includes links used to operate on that item.

Navigation Paths to items

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceIndex
   items
```

Supported Methods for items

The following methods are supported by this resource:

- `GET`

– **request - Parameters:** `startIndex`, `itemsPerPage`, `utoken`

  For information about these common parameters, see "Common Request Query Parameters".

  The following additional parameters are available:

  * `similarityURN`—the URN of the similarity calculation to be used determine which objects are similar to the recommended object. For example `item-tag`, `gs-edit`, `user-connect`
  * `srcClassURN`—the node class URN that identifies the type of the object that was used to request the recommendations. For example `WC.user`, `WC.group-space`.
  * `srcObjectURN`—the URN of the object that was used to request the recommendations. For example `monty`, `1000`.
  * `trgClassURN`—the node class URN that identifies the type of the recommended object
  * `trgObjectURN`—the object URN that provides a unique identifier for the recommended object
  * `userCredentialClassURN`—the node class URN of the user exercising the REST API. The default value is `WC.user`. If you are integrating the Activity Graph engine with another application, this may need to be a different node class.

– **response - body:** 1 or more items

For more information, see Section 54.5.2.5, "Templates."

Resource Types Linked to from items

Table 46–11 lists the resource types that the client can link to from this resource.

Table 46–11 Related Resource Types for items

rel	resourceType
self	urn:oracle:webcenter:activitygraph:items

46.3.2.4.4 urn:oracle:webcenter:activitygraph:items:item Use this resource type to identify the URI to use to update (`PUT`) a recommendation to indicate user interest in the recommended object.

Navigation Paths to item

This section shows how the client can navigate through the hypermedia to access the item resource:

```
resourceIndex
   recommendations
      recommendation
         item
resourceIndex
   items
      item
```

Supported Methods for item

The following method is supported by this resource type:

- `PUT`

- **Request—Body:** item
- **Response—Body:** item

Writable Elements for item

Table 46–12 lists the writable elements for this resource type.

Table 46–12 Writable Elements for item

Element	Type	Required	Constraints	Description
status	String	Yes	interested not interested	Whether the user is interested in the object

Read-only Elements for item

Table 46–13 lists the read-only elements for this resource type. Not all of these elements are available for all objects.

Table 46–13 Read-only Elements for item

Element	Type	Description
classURN	String	Node class of the object
objectURN	String	Identifier for the object
name	String	Name of the object
description	String	Description of the object
modified	Date	Date on which the object was last updated
modifiedByUser	PersonReference	User information about the user who last updated the object, including GUID, ID, display name, and a link to the profile icon
author	PersonReference	User information about the user who created the object, including GUID, ID, display name, and a link to the profile icon

Resource Types Linked to from item

Table 46–14 lists the resource types that the client can link to from this resource.

Table 46–14 Related Resource Types for item

rel	resourceType
self	urn:oracle:webcenter:activitygraph:items:item
via	`resourceType` of the underlying object

46.4 Extending the Activity Graph Service

Out of the box, Activity Graph includes metadata definitions for mapping WebCenter Portal service event data to WebCenter Portal's Analytics. This metadata is automatically loaded the first time the Activity Graph engines application starts.

You can extend Activity Graph metadata to change how actions are gathered from WebCenter Portal's Analytics by manipulating XML files. The XML files can be exported, edited in a text editor, and then imported.

To update Activity Graph metadata definitions, you must first export them to a local XML file:

- For information about exporting the Activity Graph metadata definitions, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- For information about exporting Analytics mapping metadata definitions, see the section "Exporting Provider Configuration" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

After exporting the metadata definitions to an appropriate file, you can then edit the file in an editor of your choice to add your own definitions. When you have made your changes, you must import the metadata file back to the managed server. For information about importing Activity Graph metadata definition files, see the section "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following subsections:

- Section 46.4.1, "Defining Custom Node Classes"
- Section 46.4.2, "Defining Custom Actions"
- Section 46.4.3, "Defining Custom Similarity Calculations"
- Section 46.4.4, "Defining Custom Rank Calculations"
- Section 46.4.5, "Registering Custom Activity Providers"
- Section 46.4.6, "Registering Custom QRPPs"

46.4.1 Defining Custom Node Classes

The Activity Graph metadata provide definitions for the node classes that represent WebCenter Portal objects, such as users, spaces, and documents. For more information and for a list of the default node classes supported by WebCenter Portal, see "Node Classes".

You can define your own node classes, for other WebCenter Portal objects or for objects from other applications, by exporting the Activity Graph metadata definitions to an XML file, editing the file, and then importing the metadata back into WebCenter Portal. For example, if you want to integrate your CRM application with Activity Graph, you could define a node class for service requests.

To define a custom node class:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to define the new node class. For each custom node class, you must specify:
   - `URN`—An attribute of the `nodeClass` element, this is a string that uniquely identifies the node class
   - `nodeType`—A sub-element that indicates whether objects defined by the node class are items or users. Valid values are `Item` or `User`.

- `numericURNs`—A sub-element that indicates whether or not the URNs of the objects of this class are numeric. This enables you to integrate Activity Graph more efficiently with applications that use numeric IDs. Valid values are `true` or `false`.
- `properties`—A list of name, value pairs. A node class's properties are available at runtime to provide additional metadata about the class. For example, all out-of-the-box WebCenter Portal objects modeled in Activity Graph define properties for `serviceID` and `resourceType`, which are used by the Activity Graph task flows to tailor the display of recommendations based on their service and resource type:

```
<properties>
  <property name="oracle.webcenter.resourceType" value="wiki" />
  <property name="oracle.webcenter.serviceID"
value="oracle.webcenter.doclib" />
</properties>
```

Example 46–1 Node Class Definition

```
<nodeClasses>
  <nodeClass URN="WC.wiki-page">
    <nodeType>Item<nodeType>
    <properties>
      <property name="oracle.webcenter.resourceType" value="wiki" />
      <property name="oracle.webcenter.serviceID" value="oracle.webcenter.doclib"
/>
    </properties>
  </nodeClass>
</nodeClasses>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.4.2 Defining Custom Actions

Activity Graph includes metadata definitions for the actions that occur in Framework applications. For more information, see "Actions".

You can define your own actions, for WebCenter Portal or for other applications, by exporting the Activity Graph metadata definitions to an XML file, editing the file, and then importing the metadata back into WebCenter Portal. For example, if you want to integrate your CRM application with Activity Graph, you could define new actions for opening a service request, assigning a service request to a customer service representative, and closing a service request.

To define a custom action:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to define the new action. For each custom action, you must specify:
   - `URN`—An attribute of the `action` element, this is a string that uniquely identifies the action

- `actionType`—A sub-element. Valid values are:
  - `Simple`—These actions are useful for counting. They have no other metadata besides the source, the target, and the occurred time. For example, WebCenter Portal comes with a preregistered `view-count` simple action whose associated relation value increases each time the person clicks on the same item, but also decays over time.
  - `Boolean`—These actions are useful when you just want to know whether something has happened or not, but not how many times it has happened. Boolean actions have one additional Boolean as metadata. For example, WebCenter Portal comes with a Boolean `view` action whose associated relation value doesn't increase each time the person clicks on the same item. It just records whether a person has clicked on a specific item or not.
  - `Non-negative Integer`—Actions that have one additional integer as metadata, where the integer cannot be negative. Rating (assigning a number of stars) is the typical example.
  - `Integer`—Actions that have one additional integer as metadata. This can be used for ratings that allow negative values.
- `symmetric`—A sub-element that indicates whether the source and target of the action are interchangeable. An example of a symmetric action is the `connect` action, which occurs when two users connect in the People Connections service. Valid values are `true` or `false`.
- `sourceType`—A sub-element that identifies the type of object that performs the action. Valid values are `User` or `Item`.
- `targetType`—A sub-element that identifies the type of object on which the action is performed. Valid values are `User` or `Item`.
- `relationType`—A sub-element. Valid values are `Sum` or `LastAssigned`. `Sum` actions increment each time they occur, for example the `edit-count` action. `LastAssigned` actions keep whatever value was passed in the most recent occurrence of the action. Non-counting simple actions like `create` and `edit` are `LastAssigned`. An example of a `LastAssigned Integer` action would be a rating action.
- `relationDecayPeriod`—(Optional) A sub-element that identifies the amount of time, in days, after which the action starts to lose (or decay) value. When computing the relation value, the value of each action is multiplied by the decay factor every decay period following the occurrence of the action.
- `relationDecayFactor`—(Optional) A sub-element that is a floating value between 0 and 1 that determines how much the action's value decreases (or decays) after the decay period. When computing the relation value, the value of each action is multiplied by the decay factor every decay period following the occurrence of the action.

Example 46–2 Action Definition

```
<actions>
  ...
  <action URN="connect">
    <actionType>Boolean</actionType>
    <symmetric>true</symmetric>
    <sourceType>User</sourceType>
    <targetType>User</targetType>
    <relationType>LastAssigned</relationType>
```

```
    <relationIsNonNegative>true</relationIsNonNegative>
  </action>
  ...
  <action URN="edit-count">
    <actionType>Simple</actionType>
    <symmetric>false</symmetric>
    <sourceType>User</sourceType>
    <targetType>Item</targetType>
    <relationType>Sum</relationType>
    <relationDecayPeriod>1</relationDecayPeriod>
    <relationDecayFactor>0.97</relationDecayFactor>
  </action>
  ...
</actions>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.4.3 Defining Custom Similarity Calculations

Activity Graph includes metadata definitions for the similarity calculations that are used by WebCenter Portal. For more information, see "Similarity Calculations".

You can define your own similarity calculations, for WebCenter Portal or for other applications, by exporting the Activity Graph metadata definitions to an XML file, editing the file, and then importing the metadata back into WebCenter Portal. For example, if you want to integrate your CRM application with Activity Graph, you could define an `item-assign` similarity calculations for to help recommend other service requests that were assigned to the same person.

To define a custom similarity calculation:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to define the new similarity calculation. For each similarity calculation, you must specify:
   - `URN`—An attribute of the similarityCalculation element that is a string that uniquely identifies the similarity calculation.
   - `similarityFunction`—A sub-element that identifies the function that measures the similarity of items. There is currently only one supported similarity function: `Tanimoto`. This function measures similarity between two items as the ratio of the number of common actions to the total number of actions on those items.
   - `domainClasses`—A sub-element that identifies the node classes for which similarity scores are calculated. These are the targets of the actions represented in the similarity calculation's relation combination.
   - `backgroundClasses`—A sub-element that identifies the node classes that are the sources of the actions in the similarity calculation's relation combination.

- `relationCombination`—A sub-element that defines a new relation by combining one or more registered relations. There are two types of relation combination:
  - `Boolean OR` (used by all WebCenter Portal out-of-the-box similarity calculations) The resulting relation combination has a value of `1` (meaning true) if any of the component relations have a positive value, and `0` (meaning false) otherwise.
  - `Weighted Sum` (used by all WebCenter Portal out-of-the-box rank calculations) The resulting relation combination is a weighted sum of the values for each of its component relations.

  For each component relation, specify:
  - `actionURN`—the URN of the action for the component relation
  - `use inverse`—set to `false` (the default) to use the component relation to calculate similarity directly for the target objects, or `true` to use the component relation to calculate similarity for the source objects rather than the target objects. For example, if the `view` relation has the source object (`user`) viewing a target object (`document`), then to calculate similarity for documents, set `use inverse=false`. To calculate similarity for source objects (users), set `use inverse=true`.
  - `weight`—(for `Weighted Sum` relation combinations) the weight to apply to the component relation represented as a floating-point number

Example 46–3 Similarity Calculation Definition

```
<similarityCalculations>
  <similarityCalculation URN="item-edit">
    <similarityFunction>Tanamoto</similarityFunction>
    <domainClasses>
      <URN>document:wcServiceID=oracle.webcenter.doclib</URN>
      <URN>wiki-page:wcServiceID=oracle.webcenter.wiki</URN>
      <URN>blog:wcServiceID=oracle.webcenter.wiki</URN>
      <URN>topic:wcServiceID=oracle.webcenter.collab.forum</URN>
    </domainClasses>
    <backgroundClasses>
      <URN>user:wcServiceID=oracle.webcenter.people</URN>
    </backgroundClasses>
    <relationCombination type="BOOLEAN_OR">
      <component inverse="false" actionURN="create" />
      <component inverse="false" actionURN="comment" />
      <component inverse="false" actionURN="edit" />
    </relationCombination>
  </similarityCalculation>
</similarityCalculations>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.4.4 Defining Custom Rank Calculations

Activity Graph includes metadata definitions for the rank calculations that are used by the Activity Graph Rank Engine to calculate the importance of nodes in the activity graph. For more information, see the section "Setting Up Activity Rank for Oracle

Secure Enterprise Search" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

You can define your own rank calculations, for WebCenter Portal or for other applications, by exporting the Activity Graph metadata definitions to an XML file, editing the file, and then importing the metadata back into WebCenter Portal.

To define a custom rank calculation:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to define the new rank calculation. For each rank calculation, you must specify:
   - `URN`—An attribute of the `rankCalculation` element, this is a string that uniquely identifies the rank calculation.
   - `domainClasses`—A sub-element that identifies a list of node classes to which the rank calculation applies.
   - `relationCombination`—A sub-element that defines a new relation by combining one or more registered relations. There are two types of relation combination:
     - `Boolean OR` (used by all WebCenter Portal out-of-the-box similarity calculations) The resulting relation combination has a value of `1` (meaning true) if any of the component relations have a positive value, and `0` (meaning false) otherwise.
     - `Weighted Sum` (used by all WebCenter Portal out-of-the-box rank calculations) The resulting relation combination is a weighted sum of the values for each of its component relations.

     For each component relation, specify:
     - `actionURN`—the URN of the action for the component relation
     - `use inverse`—set to `false` (the default) to use the component relation to calculate similarity directly for the target objects, or `true` to use the component relation to calculate similarity for the source objects rather than the target objects. For example, if the `view` relation has the source object (`user`) viewing a target object (`document`), then to calculate similarity for documents, set `use inverse=false`. To calculate similarity for source objects (users), set `use inverse=true`.
     - `weight`—(for `Weighted Sum` relation combinations) the weight to apply to the component relation represented as a floating-point number
   - `resultAcceptorClass`—A sub-element that identifies the fully qualified class name for the class that implements the `IRankResultAcceptor` interface. This class receives a set of object rankings from the Rank Engine and stores them in a search engine where they can later be used to influence search query ranking. WebCenter Portal includes one rank acceptor out of the box, which will persist ranks to Oracle Secure Enterprise Search.

Example 46–4 Rank Calculation Definition

```
<rankCalculations>
  <rankCalculation URN="activity-rank">
    <domainClasses>
```

```
      <URN>WC.user</URN>
      <URN>WC.document</URN>
      <URN>WC.wiki-page</URN>
      <URN>WC.blog</URN>
    </domainClasses>
    <relationCombination type="WEIGHTED_SUM">
      <component actionURN="connect" weight="10.0" />
      <component actionURN="edit" weight="20.0" inverse="true" />
      <component actionURN="view-count" weight="1.0" />
      <component actionURN="create" weight="100.0" inverse="true" />
      <component actionURN="create" weight="100.0" />
      <component actionURN="edit-count" weight="20.0" />
      <component actionURN="download" weight="5.0" />
      <component actionURN="tag" weight="10.0" />
      <component actionURN="comment" weight="10.0" />
    </relationCombination>
    <resultAcceptorClass>
      oracle.webcenter.activitygraph.providers.rankAcceptors.ses.SesRankResultAcceptor
    </resultAcceptorClass>
  </rankCalculation>
</rankCalculations>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.4.5 Registering Custom Activity Providers

Activity providers are used by the Activity Graph engine during the gathering process to generate activities from recorded occurrences of actions. For example, the Analytics Activity Provider reads actions from the Analytics event tables and uses a registered set of mappings to generate activities. These activities are then used to determine relations, which are used in turn to determine recommendations and search ranks.

If you want to integrate other applications with the Activity Graph engine, you can create your own activity providers to generate activities from those applications.

To make a custom activity provider available to the Activity Graph engine, you must register it by adding an activity provider assignment to the Activity Graph metadata definitions. An activity provider assignment maps the triple of the `action`, `srcClass`, and `trgClass` to the Java class that implements the activity provider.

> **Note:** If more than one triple maps to the same provider class, there must be a provider assignment for each triple. For example, the out-of-the-box metadata declares a number of provider assignments mapping all triples to a single class, `AnalyticsActivityProvider`.

To register a custom activity provider:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to define the new activity provider assignment. For each activity provider assignment, you must specify:

- `action`—An attribute of the `providerAssignment` element, this specifies which action is being mapped to the `providerClass`. Specify one registered action.
- `srcClass`—An attribute of the `providerAssignment` element, this specifies which source class is being mapped to the `providerClass`. Specify one registered node class.
- `trgClass`—An attribute of the `providerAssignment` element, this specifies which target class is being mapped to the `providerClass`. Specify one registered node class.
- `providerClass`—A sub-element that identifies the fully qualified name of the Java class that implements the activity provider for the specified action, source class, and target class triple.

Example 46–5 Activity Provider Assignment Definition

```
<providerAssignments>
  <providerAssignment action="connect" srcClass="WC.user" trgClass="WC.user">
    <providerClass>
     oracle.webcenter.activitygraph.providers.activityProviders.analytics.AnalyticsActivityProvider
    </providerClass>
  </providerAssignment>
</providerAssignments>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.4.6 Registering Custom QRPPs

WebCenter Portal provides several QRPPs to filter recommendation results. For more information, see "Query Result Post-Processors (QRPPs)".

You can create your own QRPPs to further filter recommendation results or to provide display metadata for recommendations when you are integrating the Activity Graph engine with other applications. To make a QRPP available to the Activity Graph engine, you must register it by adding a QRPP registration to the Activity Graph metadata definitions.

To register a custom QRPP:

1. Use the WLST command `exportAGMetadata` to export the Activity Graph metadata to a local XML file. For more information, see the section "Exporting Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Edit the XML file to register the new QRPP. For each QRPP, you must specify:
   - `URN`—An attribute of the `QRPP` element, this is a string that uniquely identifies the QRPP.
   - `priority`—An attribute of the `QRPP` element, this is an integer indicating the order in which the QRPP should be run in relation to the other registered QRPPs.
   - `description`—A sub-element that provides a brief description of what the QRPP does.

- `providerClass`—A sub-element that identifies the fully qualified name of the Java class that implements the QRPP.

Example 46–6 QRPP Definition

```
<QRPPs>
  <QRPP priority="1" URN="WebCenter Portal Security QRPP">
    <description>
      Uses the WebCenter Portal Resource Authorizer to perform security filtering.
    </description>
    <providerClass>
      oracle.webcenter.activitygraph.providers.qrpps.security.WCSecurityQueryResultPostProcessor
    </providerClass>
  </QRPP>
</QRPPs>
```

3. Use the WLST command `importAGMetadata` to import the updated Activity Graph metadata file to the managed server where the Framework application is deployed. For more information, see "Importing Activity Graph Metadata" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

46.5 Troubleshooting the Activity Graph Service

This section provides information to assist you in troubleshooting problems you may encounter while using the Activity Graph service.

The following troubleshooting solutions assume that:

- The Activity Graph engine is deployed correctly
- The `WC_Utilities` managed server is up and running
- The property `openusage enabled` is `true`

46.5.1 Troubleshooting the Activity Graph Service Task Flows (All Task Flows)

Problem

Recommendations are not shown in the task flows even after valid user activity.

Solution

Check whether the events are being captured by the Analytics Event Collector. You can verify this by checking the collector logs. If the logs do not show the events being captured, the problem is with the Analytics Event Collector.

For more information, see the section "Validating Analytic Event Collection" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Problem

The Analytics Event Collector logs show that events are being captured but recommendations are not shown in the task flows.

Solution

Check the last run time of the Activity Graph engine on the Activity Graph Schedule and Status Page. You may need to adjust the schedule or launch an immediate run in order to gather and analyze recent events.

Problem

The Activity Graph engine has run successfully recently, but recommendations are not shown in the task flows.

Solution

Verify the relation tables in the Activities database (`ActivitiesDS`) for the presence of any data. If the Activity Graph engines are working correctly, these relation tables should have some data in them. If data is present in the tables and there are still no recommendations in the task flows, then the task flows might be broken.

46.5.2 Troubleshooting the Recommended Connections Task Flow

Problem

Some users are not shown as recommended connections in spite of valid user interactions.

Solution

This might happen if WebCenter Portal: Spaces and the services are not wired with the same OID. In this case, users who are not present in the OID to which the services are wired will not be suggested recommendation connections. The same users should be present in both the OIDs, or the WebCenter Portal: Spaces and services should be wired to the same OID.

46.5.3 Troubleshooting the Similar Spaces Task Flow

Problem

A space is not suggested in the Similar Spaces task flow even after a successful user interaction.

Solution

Check to see that the logged in user has permissions to access the space by finding the space in the Browse Spaces task flow.

46.5.4 Troubleshooting the Similar Items Task Flow

For recommended items to be displayed in the Similar Items task flow, the item for which the recommended items are to be displayed should be selected in another task flow on the page (for example, the Document Manager task flow).

Problem

An item is not suggested in the Similar Items task flow even after valid user interaction.

Solution

This could be because the current user does not have view privileges on the item. This is the correct behavior. Users should not see recommendations for items on which they do not have sufficient privileges. If the user should be able to see the item, grant the user sufficient privileges.

Problem

Specific types of items are not being displayed (for example, documents, wikis, and blogs are seen in the task flow, but not discussions).

Solution

Check the status of the specific service. If the service is unavailable items from that service will not be displayed in the Similar Items task flow.

Problem

Items are suggested only for some users; other users do not get any suggested items in the Similar Items task flow.

Solution

This might happen if WebCenter Portal: Spaces and the services are not wired with the same OID. In this case, users who not present in the OID to which the services are wired will not see suggested items from that service. The same users should be present in both the OIDs, or the WebCenter Portal: Spaces and services should be wired to the same OID.

Part VIII

Helping Users Keep Track

Part VIII contains the following chapters:

- Chapter 47, "Integrating the Analytics Service"
- Chapter 48, "Integrating the Lists Service"
- Chapter 49, "Integrating Notifications"
- Chapter 50, "Integrating the Recent Activities Service"
- Chapter 51, "Integrating the Worklist Service"
- Chapter 52, "Integrating the Events Service"
- Chapter 53, "Integrating the RSS Service"

47

Integrating the Analytics Service

The Analytics service enables you to display usage and performance metrics for Framework applications. This chapter describes how to integrate the Analytics service into applications built using Oracle WebCenter Portal: Framework.

This chapter includes the following sections:

- Section 47.1, "Introduction to the Analytics Service"
- Section 47.2, "Basic Configuration for the Analytics Service"
- Section 47.3, "Building Analytics Reports"

For information about using any of the analytics task flows, see Chapter "Working with the Analytics Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

For a detailed list of database tables and parameters, see Appendix I, "WebCenter Portal Analytics Database Schema."

47.1 Introduction to the Analytics Service

The Analytics service offers real-time usage and activity reporting for your portal. This section provides an overview of the Analytics service, its task flows, and its usages. It includes the following subsections:

- Section 47.1.1, "Understanding the Analytics Service"
- Section 47.1.2, "Requirements for the Analytics Service"
- Section 47.1.3, "What Happens at Runtime"

47.1.1 Understanding the Analytics Service

The Analytics service allows administrators and users to track and analyze traffic and usage in WebCenter Portal application. The Analytics service provides the following basic functionality:

- **Usage Tracking Metrics**: The Analytics service collects and reports metrics for common portal functions, including community, page, portlet, and document hits.
- **Behavior Tracking**: The Analytics service can be used to analyze WebCenter Portal metrics to determine usage patterns, such as page visit duration and usage over time.
- **User Profile Correlation**: The Analytics service can be used to correlate metric information with user profile information. Usage tracking reports can be viewed and filtered by user profile data such as country, company or title.

- **Custom Reporting:** Developers and business users can build custom reports that query the analytics data. For more details, see Section 47.3.1, "Using SQL Data Controls".

Table 47–1 lists the analytics task flows available with WebCenter Portal. These task flows work similarly for Spaces and Framework applications. For detailed information about these task flows and how to use them, see "Understanding Analytics Task Flows in Spaces" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Table 47–1 Analytics Task Flows Available in JDeveloper

Task Flows	Description
WebCenter Portal Traffic	A summarized view for common events within the portal.
Page Traffic	Displays the number of page hits and the number of unique users that visited any page within the portal.
Login Metrics	Reports portal logins.
Portlet Traffic	Displays usage data for a portlet.
Portlet Response Time	Displays performance data for a portlet.
Portlet Instance Traffic	Displays usage data for a portlet instance*.
Portlet Instance Response Time	Displays performance data for a portlet instance*.
Search Metrics	Tracks portal searches.
Document Metrics	Tracks document views.
Wiki Metrics	Tracks most popular/least popular wikis.
Blog Metrics	Tracks most popular/least popular blogs.
Discussion Metrics	Tracks most popular/least popular discussions.

* If the same portlet is displayed on several different pages, each placement is known as a portlet instance.

47.1.2 Requirements for the Analytics Service

The Analytics service requires that the *analytics schema* (`ACTIVITIES`) is installed and up and running. In addition, Oracle WebCenter Portal's *Analytics Collector* must be up and running on the `WC_Utilities` managed server. For detailed installation instructions, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

On install, the Analytics Collector is configured to receive events out-of-the-box, using installation defaults. If the default values are not suitable for your installation or you want to deploy Analytics Collectors in a cluster, you may need to configure different values using WLST or Fusion Middleware Control. For more details, see "Configuring Analytics Collector Settings" *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To display usage and performance metrics through analytics task flows you must enable event generation for your Framework application and configure a connection to the analytics database (`ACTIVITIES`). For details on how to do this in JDeveloper, see Section 47.2.2, "Setting up Connections for the Analytics Service".

47.1.3 What Happens at Runtime

At runtime, user activity in your Framework application generates event data. For example, every time a user logins in, reads a discussion topic, views a document, and

so on, the event is recorded. The OpenUsage API sends event metrics to the Analytics Collector using UDP (User Datagram Protocol) and the event data is stored in the analytics database (ACTIVITIES).

In a non-clustered environment the Framework application is configured with the location of the collector, and all events are transmitted to that location.

In a clustered environment both the Framework application and Analytics Collector are configured with a cluster-specific channel name. Each collector periodically broadcasts a heartbeat with its location to the cluster-specific channel. The Framework application listens to the channel for these collector heartbeats, and when it hears one, adds the collector to its list of known collectors. When the Framework application sends an event it randomly selects a collector from its list and sends the event to that collector. If a collector stops (either being stopped purposefully or failing) it stops broadcasting a heartbeat. When the Framework application stops hearing the heartbeat it removes the collector from its list and stops sending events to that collector. If the Framework application does not hear any collector heartbeats it does not send any events.

Analytics task flows and custom analytics reports (based on SQL data controls) that are included in Framework applications display the metrics collected for standard events by querying the analytics database at runtime. For more information about the task flows at runtime, see "Working with Analytics Task Flows" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

47.2 Basic Configuration for the Analytics Service

This section describes how to include the Analytics service in your Framework applications.

This section contains the following subsections:

- Section 47.2.1, "Configuration Roadmap for the Analytics Service"
- Section 47.2.2, "Setting up Connections for the Analytics Service"
- Section 47.2.3, "Adding Analytics Event Code to Your Application"
- Section 47.2.4, "Configuring a Namespace for Analytics Customizations in MDS"
- Section 47.2.5, "Adding Analytics Task Flows at Design Time"
- Section 47.2.6, "Setting up Security for Analytics Task Flows and Usage Data"

See also, Section 47.1.2, "Requirements for the Analytics Service".

47.2.1 Configuration Roadmap for the Analytics Service

The flow chart depicted in Figure 47–1 and Table 47–2 in this section provide an overview of the prerequisites and tasks required to get the Analytics service working in Framework applications.

Figure 47–1 Configuring the Analytics Service for Framework applications

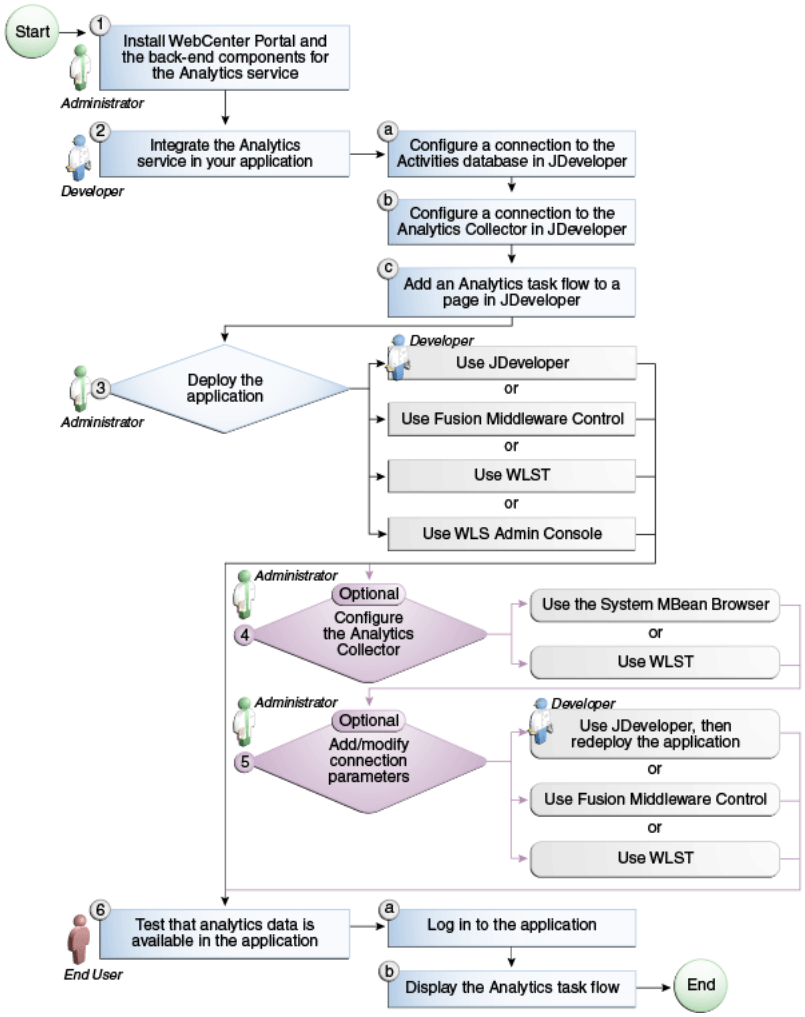


Table 47–2 Configuring the Analytics Service for Framework applications

Actor	Task	Sub-task
Administrator	**1.** Install Oracle WebCenter Portal: Framework and the back-end components for the Analytics service	
Developer	**2.** Integrate the Analytics service in your Framework application	**2.a** Configure a connection to the Activities database in JDeveloper **2.b** Configure a connection to the Analytics Collector in JDeveloper **2.c** Add an analytics task flow to a page in JDeveloper

Table 47–2 (Cont.) Configuring the Analytics Service for Framework applications

Actor	Task	Sub-task
Developer/ Administrator	**3.** Deploy the Framework application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Console (Administrator)	
Administrator	**4.** (Optional) Configure the Analytics Collector using either of the following tools: ■ System MBean Browser ■ WLST	
Developer	**5.** (Optional) Instrument events for pages and portlets using JDeveloper, then redeploy the application.	
Developer/ Administrator	**6.** (Optional) Add/modify connection parameters using one of the following tools: ■ JDeveloper, then redeploy the application (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator)	
End User	**7.** Test that analytics data is available in the Framework application	**6.a** Log in to the Framework application **6.b** Display the analytics task flow

47.2.2 Setting up Connections for the Analytics Service

The Analytics service requires that the Framework application is connected to the analytics database (`ACTIVITIES`). You must also configure the Framework application to send event information to a specific Analytics Collector.

> **Note:** While you can set up the connections to back-end servers at design time in JDeveloper, you can later add, delete, or modify connections in your deployed environment using Fusion Middleware Control. For more information, see "Managing the Analytics Service" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following subsections:

- Section 47.2.2.1, "How to Set Up a Connection to the Analytics Database"
- Section 47.2.2.2, "How to Set Analytics Collector Properties"
- Section 47.2.2.3, "How to Set Up a Connection to the Analytics Collector"

47.2.2.1 How to Set Up a Connection to the Analytics Database

The Analytics service requires a connection to the analytics database (`ACTIVITIES`) where analytics event data is stored.

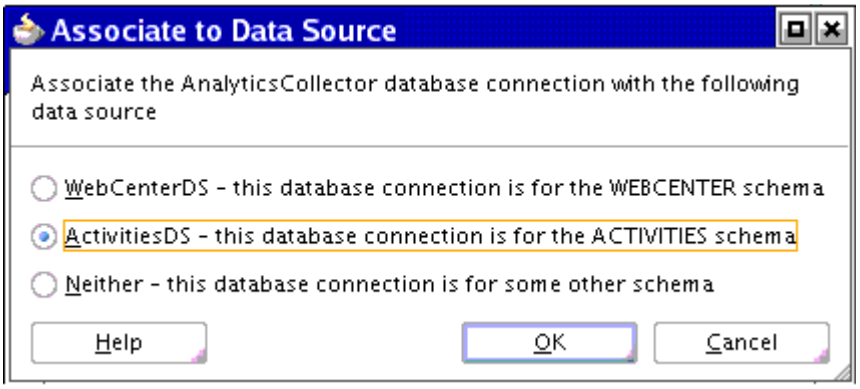
To set up the analytics database connection:

1. Ensure that the `ACTIVITIES` database is up and running.

   See Section 47.1.2, "Requirements for the Analytics Service".

2. In the Application Navigator, expand the Applications Resources panel.
3. Right-click **Connections**, then choose **New Connection**, and then **Database**.
4. In the Create Database Connection dialog, enter the following information for the database connection:
   - **Connection Name:** *`connectionName`* (for example, `myAnalyticsDatabaseConnection`
   - **Connection Type:** `Oracle (JDBC)`
   - **Username:** *`username`* (a user with access to the database)
   - **Password:** *`password`* (the specified user's password)
   - **Host Name:** *`host`* (where the analytics database is installed, for example `localhost`)
   - **JDBC Port:** *`port`* (for example, `1521`)
   - **SID:** *`sid`* (system identifier for the database)
5. Click **Test Connection**, and if it is successful, then click **OK**.
6. In the Associate to Data Source dialog, choose **ActivitiesDS** (Figure 47–2).

Figure 47–2 Connecting to the Analytics Database



   This creates a data-source to the `ACTIVITIES` schema so that you can test analytics in your Framework application at design-time.

7. Click **OK**.

   The connection appears as a node in the Application Resources pane, under Connections.

47.2.2.2 How to Set Analytics Collector Properties

Out-of-the-box, the Analytics Collector is installed on the `WC_Utilities` managed server and is configured to receive events using the following default values:

Analytics Collector Configuration	Default Value
Collector Host Name	`localhost`
Default Port	`31314`
Maximum Port Number	`31314`
Broadcast Type	`Multicast`
Clustering	`Disabled`
Cluster Name	- `null`
Cluster Broadcast Frequency	- `10 seconds`

If these defaults are not suitable for your installation, your administrator can configure suitable values using WLST or the Fusion Middleware Control. For more information, see "Configuring Analytics Collector Settings" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

47.2.2.3 How to Set Up a Connection to the Analytics Collector

The Analytics service requires a connection to the Analytics Collector that is collecting WebCenter Portal events and OpenUsage must be enabled in the Framework application.

Use the following OpenUsage JVM properties to set properties for the Analytics service:

- `oracle.wc.openusage.`**`enabled`** - Specifies whether to send analytics events raised using OpenUsage APIs to the Analytics Collector. Valid values are `true` and `false`. The default value is `false`.
- `oracle.wc.openusage.`**`unicast`** - Specifies whether events are sent to a clustered Analytics Collector in multicast mode or whether a single Analytics Collector using unicast communication is required. Valid values are `true` and `false`. The default value is `true` (unicast).
- `oracle.wc.openusage.`**`clustername`** - Name of the collector cluster or the host name of an Analytics Collector. Currently, clusters are not supported so specify only the host name here. The default value is `localhost`.
- `oracle.wc.openusage.`**`collectorport`** - Port on which the Analytics Collector listens for events. The default value is `31314`.
- `oracle.wc.openusage.`**`timeout`** Period of time (in seconds) used to determine availability of the collector service in multicast mode. The default value is 30 seconds.

To set Analytics Collector JVM properties:

1. In JDeveloper, choose **Run**, **Active Run Configuration**, and then **Manage Run Configurations** from the main menu.
2. In the Project Properties dialog, select the **Default** run configuration, and then click **Edit**.
3. In the **Launch Settings** section, add the following OpenUsage options to the **Java Options** field as follows:

```
-Doracle.wc.openusage.enabled=true
-Doracle.wc.openusage.clustername=localhost
-Doracle.wc.openusage.collectorport=31314
-Doracle.wc.openusage.unicast=true
```

```
-Doracle.wc.openusage.timeout=30
```

> **Note:** In the current release the `oracle.wc.openusage.unicast` value must be set to `true`. The Analytics Collector cannot be clustered in this release, so multicast is not supported.

Ensure that the values you provide, shown here in bold, match your Analytics Collector installation.

4. Click **OK** to save the default run configuration, and then click **OK** again.

These steps enable OpenUsage in the Integrated WebLogic Server for the current Framework application. If the Integrated WebLogic Server is currently running you must restart it to pick up the new settings. See also, Section 70.2, "Deploying a Framework Application to the Integrated WebLogic Server".

47.2.3 Adding Analytics Event Code to Your Application

Most events are configured out-of-the-box, so no additional coding is required in your application to send events. The only exception is page view events, which require additional code, as described in Section 47.2.3.1, "Including Event Code for Page Views."

47.2.3.1 Including Event Code for Page Views

To send page events, you must add event code to each page. Following is an example using a client JavaScript event on the page which, in turn, calls the Java Analytics API to send the actual event. To implement this example, you add the following code just below the `<af:document>` tag, replacing the `pageName` value with the name of the page for which you are sending events:

```
  <af:resource type="javascript">
       function initPageLoadEvent(event {)
            AdfCustomEvent.queue(event.getSource(),
                 "generatePageEvent",
                 {pageName:"Page and Login Statistics"},
                 true);
            event.cancel();
       }
  </af:resource>
<af:clientListener method="initPageLoadEvent" type="load"/>
<af:serverListener type="generatePageEvent"
                   method="#{AnalyticsInstrumentation.sendPageEvent}"/>
```

The example above generates a client event when the ADF document gets loaded, then the defined server listener actually sends the event. Here is the java bean that sends the event:

```
public void sendPageEvent (ClientEvent event) {
     // Checks whether Analytics events must be sent
     if (!AnalyticsUtil.isSendingEvents())
         return;

     HTTPServletRequest request = (HTTPServletRequest)
FacesContext.getCurrentInstance().getExternalContext().getRequest();
     String requestUser = request.getRemoteUser = request.getRemoteUser();

     //SEND PAGE VIEW EVENT
```

```
     String pageName = (String)event.getParameters().get("pageName");
     if(pageName == null || pageName.isEmpty() || requestUser == null)
        return;

     AnalyticsUtil.sendPageViewEvent(FacesContext.getCurrentInstance().
getViewRoot..getViewId(), //viewID
                                     "PortalApp", //spaceDisplayName
                                     requestUser, //username
                                     0,
                                     pageName,
                                     false,
                                     request);
```

47.2.4 Configuring a Namespace for Analytics Customizations in MDS

Analytics task flows use MDS to store customizations made by the user and these customizations are stored in an MDS namespace specific to analytics. If you want to enable user customizations for analytics task flows in your Framework application, you must configure a namespace for the analytics metadata in `adf-config.xml`.

> **Note:** In addition, each task flow must specify a unique MDS document in which to its store user customizations details. See Section 47.2.5.4, "How to Allow End Users to Customize Analytics Task Flows at Runtime".

To configure a namespace for analytics metadata:

1. Open `adf-config.xml`.

   Use the Application Resources panel to navigate to this file. The file is located in the `Descriptors\ADF META-INF` folder.

2. Under the `<metadata-namespaces>` element, add the following XML fragment:

   ```
   <namespace path="/oracle/webcenter/analytics/scopedMD"
   metadata-store-usage="WebCenterFileMetadataStore"/>
   ```

3. Save the file.

See also, Chapter 23, "Performing Composer-Specific MDS Configurations".

47.2.5 Adding Analytics Task Flows at Design Time

This section describes the Analytics service task flows and how to add analytics task flows to your application. It includes the following subsections:

- Section 47.2.5.1, "Analytics Task Flows"
- Section 47.2.5.2, "How to Add Analytics Task Flows to a Page"
- Section 47.2.5.3, "How to Modify Analytics Task Flow Parameters"
- Section 47.2.5.4, "How to Allow End Users to Customize Analytics Task Flows at Runtime"
- Section 47.2.5.5, "Analytics Task Flows and Task Flow Parameters"

47.2.5.1 Analytics Task Flows

Table 47–3 lists the analytics task flows available with WebCenter Portal. For detailed information about these task flows and the type of information that users can see at runtime, see "Understanding Analytics Task Flows in Spaces" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Table 47–3 Analytics Task Flows Available in JDeveloper

Analytics Task Flows	Description
WebCenter Portal Traffic	A summarized view for common events within the portal.
Page Traffic	Displays the number of page hits and the number of unique users that visited any page within the portal.
Login Metrics	Reports portal logins.
Portlet Traffic	Displays usage data for a portlet.
Portlet Response Time	Displays performance data for a portlet.
Portlet Instance Traffic	Displays usage data for a portlet instance*.
Portlet Instance Response Time	Displays performance data for a portlet instance*.
Search Metrics	Tracks portal searches.
Document Metrics	Tracks document views.
Wiki Metrics	Tracks most popular/least popular wikis.
Blog Metrics	Tracks most popular/least popular blogs.
Discussion Metrics	Tracks most popular/least popular discussions.

47.2.5.2 How to Add Analytics Task Flows to a Page

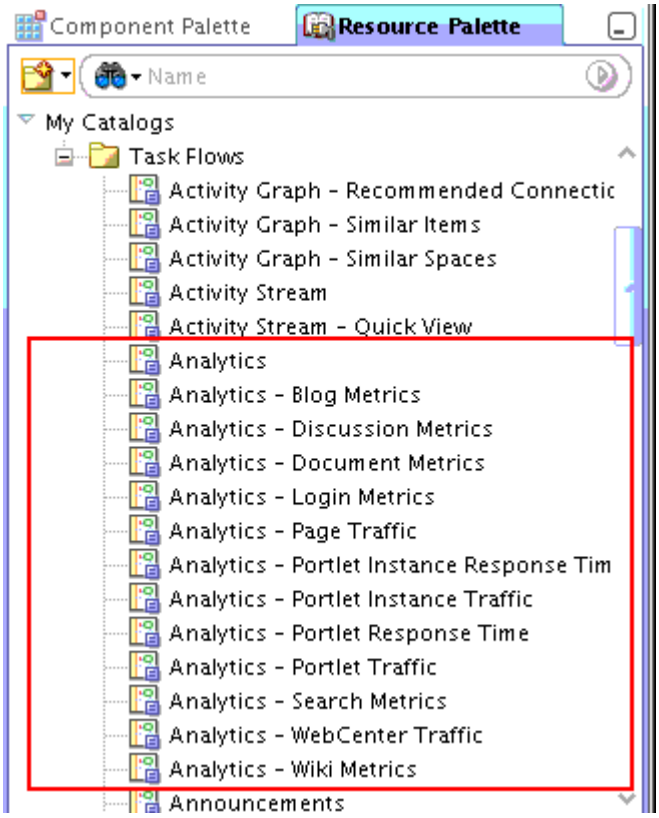
To add an analytics task flow to your Framework application:

1. Prepare your application as described in Section 47.1.2, "Requirements for the Analytics Service".
2. Ensure that your application includes the services for which analytics event data is available, that is, portlets, discussions, documents, and search.

   Note: If no usage data exists when you run the application, analytics task flows displays the message: `No data to display`

3. Open the JSF page (.jspx) on which you want to add the task flow.
4. In the Resource Palette, open the **WebCenter Portal Services Catalog**, then open the **Task Flows** folder (Figure 47–3).

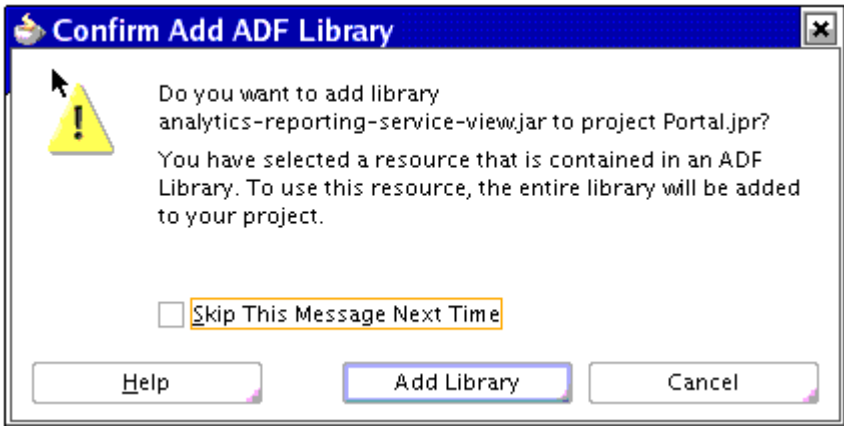
Figure 47–3 Analytics Task Flows Available Through the Resource Palette



5. Drag and drop an **Analytics** task flow onto your page inside the `<af:form>` tag.
6. Choose **Region** from the Create context menu.

   You may be prompted to add the ADF library for the Analytics service (`analytics-reporting-service-view.jar`) to the project. Confirm by clicking **Add Library** (Figure 47–4).

Figure 47–4 Adding the ADF Library for the Analytics Service



7. In the **Edit Task Flow Binding** dialog, enter values for analytics task flow parameters:
   - `applicationName` - Required

- `analyticsResourceId` - Required
- `analyticsReportTitle` - Optional
- `maxDataPointsPerSeries` - Optional

For more information, see Section 47.2.5.5, "Analytics Task Flows and Task Flow Parameters".

8. Click **OK**.

   The task flow is added to the page, and the ViewController project's libraries are configured to run the task flow.

9. Save your project and run the page to see the task flow running in the Framework application.

   If no usage data exists when you run the application, analytics task flows display the message: `No data to display`

   You may create some data for the analytics task flow to display by performing actions that relate to the task flow you are testing. For example, to test discussion metrics, create and view discussions, to test page metrics, create and view pages, and so on.

47.2.5.3 How to Modify Analytics Task Flow Parameters

Each analytics task flow has a set of required and optional task flow binding parameters. Required parameters are not mandatory but enable you to capture information that is essential to the task flow's successful function. For example, if you want user customizations for a particular task flow instance to be stored in MDS you must specify the MDS document required.

In addition, you can use task flow binding parameters to customize the appearance and behavior of a task flow instance. For example, you can use parameter values to specify a display title above your analytics data.

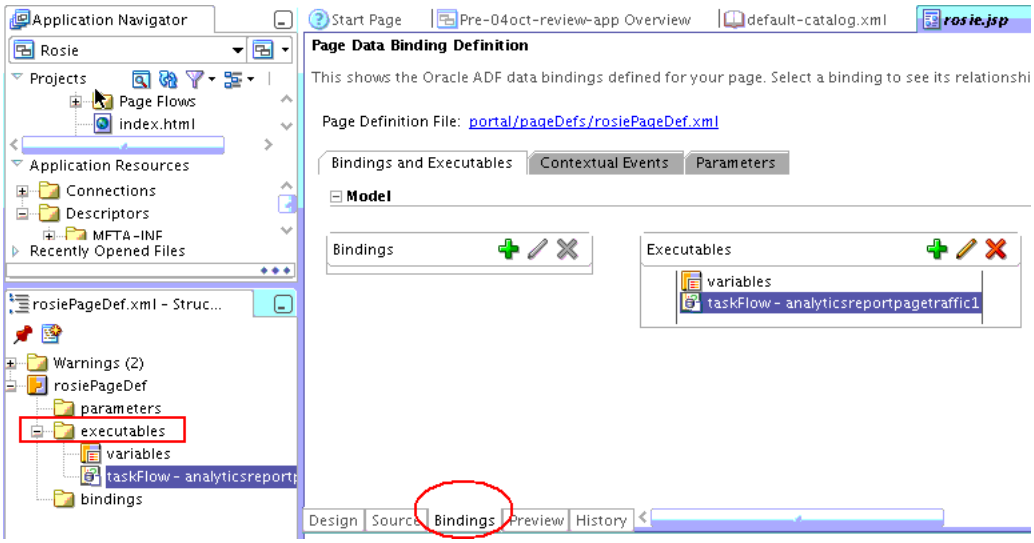
You can provide task flow binding parameter values when you drag and drop a task flow onto an application page. Doing so opens the Task Flow Bindings dialog (for more information, see Section 47.2.5.2, "How to Add Analytics Task Flows to a Page").

You can also adjust task flow binding parameter values after you have placed a task flow on a page at run time. For details, see "Setting Analytics Task Flow Properties" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

To access the Edit Task Flow Binding dialog:

1. Click the **Bindings** tab at the bottom of the page (next to the **Source** tab) to go to the Bindings view.
2. Under **Executables**, you will see a list of task flows added to the page. Select the analytics task flow (Figure 47–5).

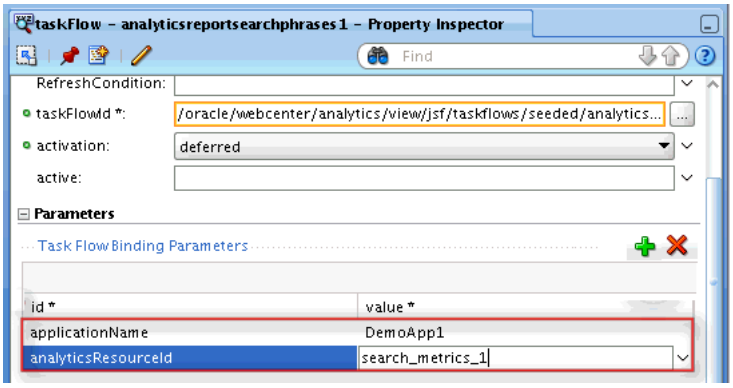
Figure 47–5 Accessing Input Parameters for Analytics Task Flows



3. Next to the Executables heading, click the **Edit selected element** icon (a pencil).
4. In the Edit Task Flow Binding dialog (Figure 47–6), specify the binding parameter values as required

   For more information, see Section 47.2.5.5, "Analytics Task Flows and Task Flow Parameters".

Figure 47–6 Adding Binding Parameters for Analytics Task Flows



5. When you are finished, save the page and run your page to see the results.

47.2.5.4 How to Allow End Users to Customize Analytics Task Flows at Runtime

If you want to enable user customizations for a particular task flow instance you must specify the MDS document where customizations are stored using the `Analytics Resource Id` parameter.

The ID must be unique, so consider using a consistent naming pattern such as `<app_name>_<page_name>_<task_flow>_<sequence>`. For example, set the `analyticsResourceId` parameter for a Page Traffic task flow to `myapp_analyticspage_pagetraffic_1`.

For details, see Section 47.2.5.3, "How to Modify Analytics Task Flow Parameters".

47.2.5.5 Analytics Task Flows and Task Flow Parameters

Table 47–4 lists and describes task flow binding parameters applicable to the Analytics service.

Table 47–4 Analytics Task Flow Binding Parameters

Parameter (* = required)	Task Flows	Description
`analyticsReportTitle`	All	Specifies the display title that appears above the analytics data, that is, you can override the default report title. See also, Figure 47–7, "analyticsReportTitle - Example".
`analyticsResourceId*`	All	Specifies the MDS document that will be generated to store user customizations/application customizations for the task flow instance in MDS. For example: `mymainloginmetrics`
`applicationName*`	All	Specifies the name of the Framework application for which you want to display analytics data. For example: `MyPortalApplication` The analytics database can be used to store event data from multiple applications so this parameter is required to identify which application data to display. If omitted, the task flow display analytics data for all applications.
`maxDataPointsPerSeries`	All	Indicates the maximum number of data points to be displayed in a bar or line chart. Enter a value between 1 and 1000. The default value is 25. Increasing the number of data points can increase the time it takes to display the chart. See also, Figure 47–8, "maxDataPointsPerSeries - Example".

Figure 47–7 analyticsReportTitle - Example

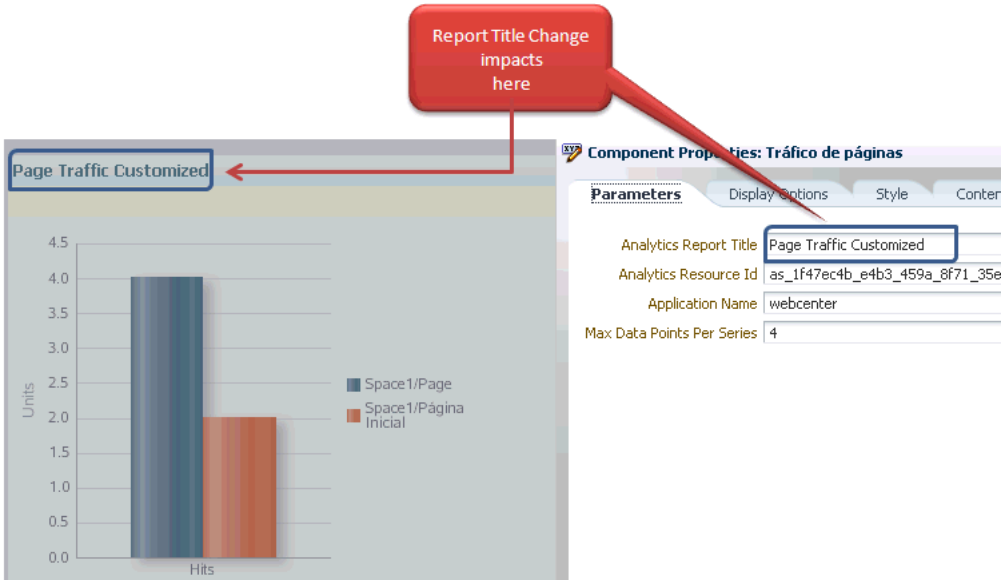
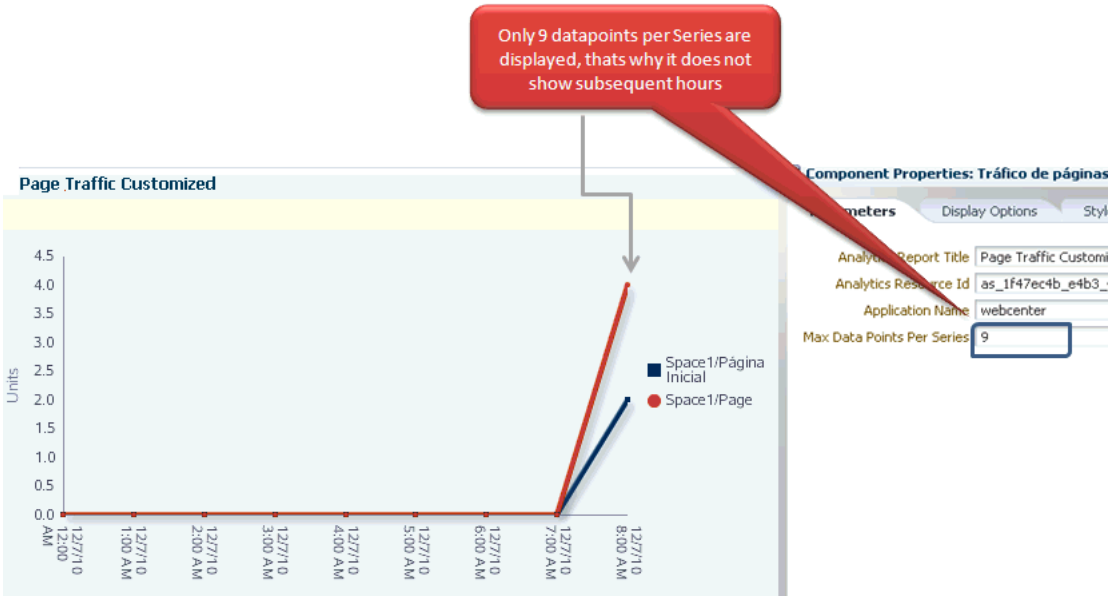


Figure 47–8 maxDataPointsPerSeries - Example



47.2.6 Setting up Security for Analytics Task Flows and Usage Data

Analytics task flows are intended to make usage metrics visible to a limited set of administrative users who perform particular business functions, such as capacity planning, quality of service (QoS) analysis, return on investment (ROI) analysis, "best bet" customization for Search, and so on.

Analytics usage data is valuable for portal analysis but might be regarded as private or sensitive to portal users. For example, the Search, Document, and Portlet reports can be configured to display activity metrics for a particular user, based on user properties such as E-mail Address, First Name, or Last Name.

To protect security and privacy interests associated with analytics task flows and custom reports:

- Manage administrative access to the analytics task flows, custom reports, and any page that displays sensitive usage data.

  To ensure that only a limited number of administrative users can add analytics task flows to pages, create reports based on custom analytics data controls, or view pages set up to display sensitive usage metrics, create a new administrative group and manage group membership accordingly.

- Manage user access to analytics task flows.

  Ensure analytics task flows and custom reports do not contain private or sensitive data unless such a view is particularly intended. If the metrics in the report do contain private or sensitive data, configure security so that only appropriate, specified users have access to the task flow or the page.

  For example, at design-time, developers can expose the analytics task flows to non-admin users by granting them appropriate privileges to the page. In addition, developers can customize the reports pre-deployment, to hide or show and predefine certain report options (such as time frame, chart type, user property filter, group by option, and so on). Administrators will be able to perform the same tasks at runtime, that is, grant page access and customize the information that

displays. Non-admin users can still personalize the reports they are allowed to see, but they cannot change customizations made by the administrator.

- Ensure that unauthenticated users are never allowed to add analytics task flows to pages or see sensitive data.

47.3 Building Analytics Reports

Out-of-the-box analytics task flows present common analytics event metrics in a specific display format. If you want to present analytics data in a different way or display custom event data, you can build a custom analytics report using SQL.

This section contains the following subsections:

- Section 47.3.1, "Using SQL Data Controls"
- Section 47.3.2, "SQL Statements for Out-of-the-Box Analytics Reports"
- Section 47.3.3, "Sample Queries for User Metrics"
- Section 47.3.4, "Sample Analytics Database Queries for Specific Metrics"

For a detailed list of database tables and parameters, see Appendix I, "WebCenter Portal Analytics Database Schema."

47.3.1 Using SQL Data Controls

Use SQL data controls to define, in an SQL query statement, the information you want to retrieve from the analytics database. When you expose data controls on a page you can choose whether the analytics data presents in a graph, table or a form, and you can also configure the bind parameters and other display options.

You can use JDeveloper to build SQL data controls at design-time. For detailed information about data controls, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

You can also create data controls in a running Framework application if runtime resource management features are enabled. To do this in the Spaces application, see "Creating a SQL Data Control" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. To create a data control in other Framework applications, see "Managing Application Resources" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Use a SQL data control to fetch data from the analytics database and display analytics data in your Framework application. Figure 47–9 shows the sample SQL statement below in a Create/Edit SQL Data Control dialog in Spaces. A database connection to the analytics database (`ActivitiesDS`) is required, and you must provide a valid database password.

Sample SQL: Page Hits by Day

```
SELECT space.name_, page.name_, space.id, page.id, count(1), fact.page_,
page.resourceid_
  FROM  asfact_wc_pagevie_0 fact, asdim_wc_groupsp_0 space, asdim_wc_pages_0 page,
asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and page.id = fact.page_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
```

```
    and page.personal_ = :ispersonal
    and space.name_ is not null
    and page.name_ is not null
    and space.id is not null
    and page.id is not null
    and fact.page_ is not null
    and page.resourceid_ is not null
GROUP BY space.name_, page.name_, space.id, page.id, fact.page_, page.resourceid_
ORDER BY count(1) desc
```

Figure 47–9 Creating SQL Data Control Dialog in Spaces



You can add as many bind parameters as required. When you add parameters, you can restrict the data retrieved from the data source based on the parameter values you specify. As this example is for the Spaces application, the name of the application is set to `webcenter` and personal pages (pages in the Home space) are included in the report shown in Figure 47–10.

Figure 47–10 Setting Report Parameters

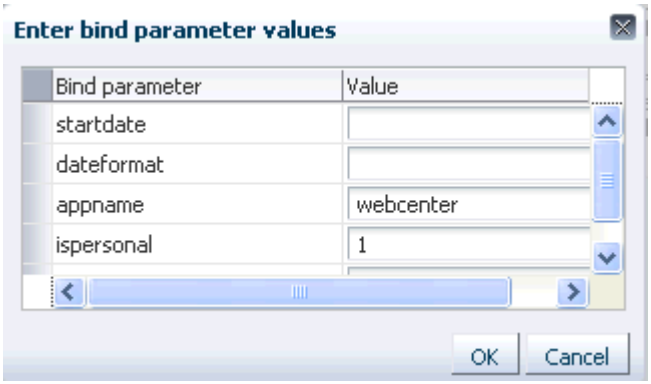
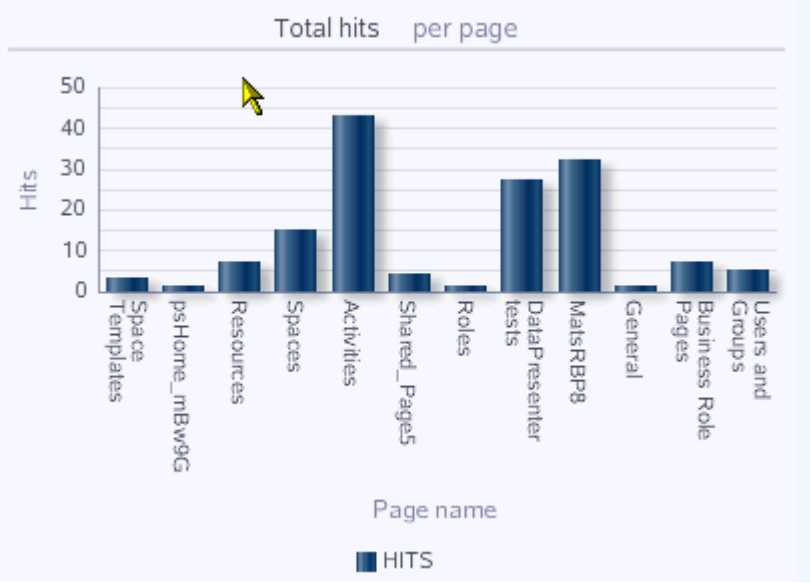


Table 47–5 Report Parameters

Parameter	Default Value	Description
`appname`	`webcenter`	The corresponding application name, always `webcenter` for Spaces. Other Framework application will have a different name.
`ispersonal`	`0 or 1`	For Spaces, determines whether or not pages created in the Home space are included in the report.
`dateformat`	`mm/dd/yyyy hh24`	Any valid pattern for dates in an Oracle database. Every component of the pattern is optional since it depends on the data you want to retrieve.
`startdate`	`01/01/2010`	Determines a start date for the data required.
`enddate`	`01/01/2011`	Determines an end date for the data required.

Figure 47–9 shows the custom report displayed as a graph.

Figure 47–11 Sample Analytics Report - Page Hits by Day



47.3.2 SQL Statements for Out-of-the-Box Analytics Reports

This section provides SQL statements that can be used as a starting point for custom reports. You can customize these statements in many ways by providing additional filters, groupings, and so on.

- Analytics SQL: WebCenter Portal Traffic
- Analytics SQL: Page Traffic
- Analytics SQL: Login Metrics
- Analytics SQL: Space Traffic
- Analytics SQL: Space Response Time
- Analytics SQL: Portlet Traffic
- Analytics SQL: Portlet Instance Traffic

- Analytics SQL: Portlet Response Time
- Analytics SQL: Portlet Instance Response Time
- Analytics SQL: Search Metrics
- Analytics SQL: Document Metrics
- Analytics SQL: Wiki Metrics
- Analytics SQL: Blog Metrics
- Analytics SQL: Discussion Metrics

For a detailed list of database tables and parameters, see Appendix I, "WebCenter Portal Analytics Database Schema."

47.3.2.1 Analytics SQL: WebCenter Portal Traffic

```
SELECT 'Spaces' Name, count(1) Hits
  FROM asfact_wc_groupsp_0 fact, asdim_wc_groupsp_0 space, asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname and space.personal_ = :ispersonal
UNION ALL
  SELECT 'Pages' Name, count(1) Hits
  FROM asfact_wc_pagevie_0 fact , asdim_wc_pages_0 page , asdim_wc_applica_0 app
  WHERE page.id = fact.page_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and page.personal_ = :ispersonal
UNION ALL
  SELECT 'Portlets' Name, count(1) Hits
  FROM asfact_wc_portlet_0 fact, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
UNION ALL
  SELECT 'Logins' Name, count(1) Hits
  FROM asfact_wc_logins_0 fact , asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
UNION ALL
  SELECT 'Searches' Name, count(1) Hits
  FROM asfact_wc_searche_0 fact , asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
UNION ALL
  SELECT 'Wikis' Name, count(1) Hits
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat) and app.name_ = :appname
```

```
    and doc.objecttype_ like '%WIKI%'
UNION ALL
  SELECT 'Blogs' Name, count(1) Hits
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and doc.objecttype_ like '%BLOG%'
UNION ALL
  SELECT 'Documents' Name, count(1) Hits
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and doc.objecttype_ like '%DOCUMENT%'
UNION ALL
  SELECT 'Discussions' Name, count(1) Hits
  FROM asfact_wc_discuss_1 fact, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
```

47.3.2.2 Analytics SQL: Page Traffic

```
SELECT  space.name_, page.name_, space.id, page.id, count(1), fact.page_,
page.resourceid_
  FROM asfact_wc_pagevie_0 fact, asdim_wc_groupsp_0 space, asdim_wc_pages_0 page,
asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and page.id = fact.page_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and page.personal_ = :ispersonal
    and space.name_ is not null
    and page.name_ is not null
    and space.id is not null
    and page.id is not null
    and fact.page_ is not null
    and page.resourceid_ is not null
GROUP BY space.name_, page.name_, space.id, page.id, fact.page_, page.resourceid_
ORDER BY count(1) desc
```

47.3.2.3 Analytics SQL: Login Metrics

```
SELECT  count(1)
 FROM asfact_wc_logins_0 fact, asdim_wc_applica_0 app
 WHERE app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
```

47.3.2.4 Analytics SQL: Space Traffic

```
SELECT  space.name_, space.id, count(1), fact.groupspace_, space.resourceid_
```

```
  FROM asfact_wc_groupsp_0 fact, asdim_wc_groupsp_0 space, asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and space.personal_ = :ispersonal
    and space.name_ is not null
    and space.id is not null
    and fact.groupspace_ is not null
    and space.resourceid_ is not null
GROUP BY space.name_, space.id, fact.groupspace_, space.resourceid_
ORDER BY count(1) desc
```

47.3.2.5 Analytics SQL: Space Response Time

```
SELECT  space.name_, space.id, avg(fact.response_time_), fact.groupspace_,
space.resourceid_
  FROM asfact_wc_groupsp_0 fact, asdim_wc_groupsp_0 space, asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and app.id = fact.application_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and space.personal_ = :ispersonal
    and space.name_ is not null
    and space.id is not null
    and fact.groupspace_ is not null
    and space.resourceid_ is not null
GROUP BY space.name_, space.id, fact.groupspace_, space.resourceid_
```

47.3.2.6 Analytics SQL: Portlet Traffic

```
SELECT  portlet.default_title_, portlet.id, count(1), fact.portlet_,
portlet.resourceid_
  FROM asfact_wc_portlet_0 fact, asdim_wc_portlet_0 portlet, asdim_wc_applica_0
app
  WHERE app.id = fact.application_
    and portlet.id = fact.portlet_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and portlet.default_title_ is not null
    and portlet.id is not null
    and fact.portlet_ is not null
    and portlet.resourceid_ is not null
GROUP BY portlet.default_title_, portlet.id, fact.portlet_, portlet.resourceid_
```

47.3.2.7 Analytics SQL: Portlet Instance Traffic

```
SELECT  space.name_, space.id, page.name_, page.id, portletinst.title_,
portletinst.id, count(1), fact.portlet_instance_, portletinst.resourceid_
  FROM asfact_wc_portlet_0 fact, asdim_wc_groupsp_0 space, asdim_wc_pages_0 page,
asdim_wc_portlet_1 portletinst, asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and page.id = fact.page_
    and app.id = fact.application_
    and portletinst.id = fact.portlet_instance_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
```

```
    and space.name_ is not null
    and space.id is not null
    and page.name_ is not null
    and page.id is not null
    and portletinst.title_ is not null
    and portletinst.id is not null
    and fact.portlet_instance_ is not null
    and portletinst.resourceid_ is not null
GROUP BY space.name_, space.id, page.name_, page.id, portletinst.title_,
portletinst.id, fact.portlet_instance_, portletinst.resourceid_
```

47.3.2.8 Analytics SQL: Portlet Response Time

```
SELECT  portlet.default_title_, portlet.id, avg(fact.response_time_),
fact.portlet_, portlet.resourceid_
  FROM asfact_wc_portlet_0 fact, asdim_wc_portlet_0 portlet, asdim_wc_applica_0
app
  WHERE app.id = fact.application_
    and portlet.id = fact.portlet_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and portlet.default_title_ is not null
    and portlet.id is not null
    and fact.portlet_ is not null
    and portlet.resourceid_ is not null
GROUP BY portlet.default_title_, portlet.id, fact.portlet_, portlet.resourceid_
```

47.3.2.9 Analytics SQL: Portlet Instance Response Time

```
SELECT  space.name_, space.id, page.name_, page.id, portletinst.title_,
portletinst.id, avg(fact.response_time_), fact.portlet_instance_,
portletinst.resourceid_
  FROM asfact_wc_portlet_0 fact, asdim_wc_groupsp_0 space, asdim_wc_pages_0 page,
asdim_wc_portlet_1 portletinst, asdim_wc_applica_0 app
  WHERE space.id = fact.groupspace_
    and page.id = fact.page_
    and app.id = fact.application_
    and portletinst.id = fact.portlet_instance_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and space.name_ is not null
    and space.id is not null
    and page.name_ is not null
    and page.id is not null
    and portletinst.title_ is not null
    and portletinst.id is not null
    and fact.portlet_instance_ is not null
    and portletinst.resourceid_ is not null
GROUP BY space.name_, space.id, page.name_, page.id, portletinst.title_,
portletinst.id, fact.portlet_instance_, portletinst.resourceid_
```

47.3.2.10 Analytics SQL: Search Metrics

```
SELECT  search.phrase_, search.id, count(1)
  FROM asfact_wc_searche_0 fact, asdim_wc_applica_0 app, asdim_wc_searche_0 search
  WHERE app.id = fact.application_
    and search.id = fact.searched_phrase_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
```

```
    and app.name_ = :appname
    and search.phrase_ is not null
    and search.id is not null
GROUP BY search.phrase_, search.id
```

47.3.2.11 Analytics SQL: Document Metrics

```
SELECT  doc.name_, doc.id, count(1), fact.document_, doc.resourceid_
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and doc.objecttype_ like '%DOCUMENT%'
    and doc.name_ is not null
    and doc.id is not null
    and fact.document_ is not null
    and doc.resourceid_ is not null
GROUP BY doc.name_, doc.id, fact.document_, doc.resourceid
```

47.3.2.12 Analytics SQL: Wiki Metrics

```
SELECT  doc.name_, doc.id, count(1), fact.document_, doc.resourceid_
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and doc.objecttype_ like '%WIKI%'
    and doc.name_ is not null
    and doc.id is not null
    and fact.document_ is not null
    and doc.resourceid_ is not null
GROUP BY doc.name_, doc.id, fact.document_, doc.resourceid_
```

47.3.2.13 Analytics SQL: Blog Metrics

```
SELECT  doc.name_, doc.id, count(1), fact.document_, doc.resourceid_
  FROM asfact_wc_doclib__0 fact, asdim_wc_documen_0 doc, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and fact.document_ = doc.id
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
    and app.name_ = :appname
    and doc.objecttype_ like '%BLOG%'
    and doc.name_ is not null
    and doc.id is not null
    and fact.document_ is not null
    and doc.resourceid_ is not null
GROUP BY doc.name_, doc.id, fact.document_, doc.resourceid_
```

47.3.2.14 Analytics SQL: Discussion Metrics

```
SELECT  forum.name_, forum.id, count(1), fact.topic_, forum.resourceid_
  FROM asfact_wc_discuss_0 fact, asdim_wc_discuss_0 forum, asdim_wc_applica_0 app
  WHERE app.id = fact.application_
    and forum.id = fact.topic_
    and fact.occurred between to_date(:startdate, :dateformat)
    and to_date(:enddate, :dateformat)
```

```
    and app.name_ = :appname
    and forum.name_ is not null
    and forum.id is not null
    and fact.topic_ is not null
    and forum.resourceid_ is not null
GROUP BY forum.name_, forum.id, fact.topic_, forum.resourceid_
```

47.3.3 Sample Queries for User Metrics

This section provides example queries for user-specific metrics. It includes the following samples:

- Sample SQL: Filter by User Property
- Sample SQL: Group by User Property

For a detailed list of database tables and parameters and information on user properties, seeSection I.5, "Analytics User Properties" in Appendix I, "WebCenter Portal Analytics Database Schema.".

47.3.3.1 Sample SQL: Filter by User Property

The following query returns the names of the pages accessed by users in the Sales department, and the number of views for each of those pages.

```
SELECT p.name_ as page, count(*) as views
  FROM asfact_wc_pagevie_0 f
    JOIN asdim_wc_pages_0 p on f.page_ = p.id
    JOIN asdim_userpropertyvalues pv on f.userid = pv.userid
    JOIN asdim_userproperties pr on pv.propertyid = pr.id
  WHERE pr.name = 'DEPARTMENT'
    and pv.value = 'Sales'
GROUP BY p.name_
```

47.3.3.2 Sample SQL: Group by User Property

The following query returns the total number of page views broken down into departments.

```
SELECT pv.value as department, count(*) as views
  FROM asfact_wc_pagevie_0 f
    JOIN asdim_userpropertyvalues pv on f.userid = pv.userid
    JOIN asdim_userproperties pr on pv.propertyid = pr.id
  WHERE pr.name = 'DEPARTMENT'
GROUP BY pv.value
```

47.3.4 Sample Analytics Database Queries for Specific Metrics

This section provides example queries for specific metrics. It includes the following samples:

- Sample SQL: User Activities
- Sample SQL: Space Activities
- Sample SQL: Space Activities for a Specific Time Period
- Sample SQL: Activity for a Service During a Specific Time Period
- Sample SQL: Search Phrases
- Sample SQL: Page Views

For a detailed list of database tables and parameters, see Appendix I, "WebCenter Portal Analytics Database Schema."

47.3.4.1 Sample SQL: User Activities

The following query returns a list of all the activities executed by a specific user during a specific time period.

```
SELECT 'Logins' AS Activity, count(*) AS Events
  FROM asfact_wc_logins_0 f
    JOIN asdim_time t ON f.timeid = t.id
    JOIN asdim_users u ON f.userid = u.id
  WHERE u.userid = 'user1'
    AND f.timeid >= (SELECT id
                     FROM asdim_time
                     WHERE year = 2011
                       AND monthofyear = 0
                       AND dayofmonth = 1
                       AND hourofday = 0)
    AND f.timeid <  (SELECT id
                     FROM asdim_time
                     WHERE year = 2011
                       AND monthofyear = 1
                       AND dayofmonth = 1
                       AND hourofday = 0)
UNION ALL
  SELECT 'Searches' AS Activity, count(*) AS Events
  FROM asfact_wc_searche_0 f
    JOIN asdim_time t ON f.timeid = t.id
    JOIN asdim_users u ON f.userid = u.id
  WHERE u.userid = 'user1'
    AND f.timeid >= (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 0
                       and dayofmonth = 1
                       and hourofday = 0)
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
UNION ALL
   ...
```

Sample Report Output

```
Activity   Events
---------------
Logins        25
Searches      15
...
```

47.3.4.2 Sample SQL: Space Activities

The following query returns a list of all activities executed in a specific space.

```
SELECT g.name_ AS Space, 'Page views' AS Activity,
    count(*) AS Events
  FROM asfact_wc_pagevie_0 f
    JOIN asdim_time t ON f.timeid = t.id
```

```
    JOIN asdim_wc_groupsp_0 g ON f.groupspace_ = g.id
  WHERE g.name_ IN ('Project1', 'Project2')
    AND f.timeid >= (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 0
                       and dayofmonth = 1
                       and hourofday = 0)
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
  GROUP BY g.name_
UNION ALL
  SELECT g.name_ AS Space,
    'Portlet views' AS Activity, count(*) AS Events
  FROM asfact_wc_portlet_0 f
    JOIN asdim_time t ON f.timeid = t.id
    JOIN asdim_wc_groupsp_0 g ON f.groupspace_ = g.id
  WHERE g.name_ in ('Project1', 'Project2')
    AND f.timeid >= (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 0
                       and dayofmonth = 1
                       and hourofday = 0)
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
  GROUP BY g.name_
```

Sample Report Output

```
Space       Activity      Events
--------------------------------
Project1    PageViews         34
Project2    PageViews         42
Project1    PortletViews      98
Project2    PortletViews      74
```

47.3.4.3 Sample SQL: Space Activities for a Specific Time Period

The following query returns a list of all activities executed in a specific space during a specific time period.

```
SELECT 'Page views' AS Activity, count(*) AS Events
  FROM asfact_wc_pagevie_0 f
    JOIN asdim_time t ON f.timeid = t.id
    JOIN asdim_wc_groupsp_0 g ON f.groupspace_ = g.id
  WHERE g.name_ = 'Project1'
    AND f.timeid >= (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 0
                       and dayofmonth = 1
                       and hourofday = 0)
```

```
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
UNION ALL
  SELECT 'Portlet views' AS Activity, count(*) AS Events
  FROM asfact_wc_portlet_0 f
    JOIN asdim_time t ON f.timeid = t.id
    JOIN asdim_wc_groupsp_0 g ON f.groupspace_ = g.id
  WHERE g.name_ = 'Project1'
    AND f.timeid >= (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 0
                       and dayofmonth = 1
                       and hourofday = 0)
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
UNION ALL
   ...
```

Sample Report Output

```
Activity       Events
---------------------
Page views         15
Portlet views     200
```

47.3.4.4 Sample SQL: Activity for a Service During a Specific Time Period

The following query returns activity for a specific service (in this example, the login service) during a specific time period and groups results by user.

```
SELECT u.userid AS "User", count(*) AS "Logins"
FROM asfact_wc_logins_0 f
  JOIN asdim_time t ON f.timeid = t.id
  JOIN asdim_users u ON f.userid = u.id
WHERE f.timeid >= (select id
                   from asdim_time
                   where year = 2011
                     and monthofyear = 0
                     and dayofmonth = 1
                     and hourofday = 0)
    AND f.timeid <  (select id
                     from asdim_time
                     where year = 2011
                       and monthofyear = 1
                       and dayofmonth = 1
                       and hourofday = 0)
GROUP BY u.userid
ORDER BY "Logins" DESC
```

Sample Report Output

```
User      Logins
--------------
```

```
user3     245
user1     240
user2     193
```

47.3.4.5 Sample SQL: Search Phrases

The following query returns the phrases searched during a specific time and lists how many times each search was executed.

```
SELECT s.phrase_ AS "Search Phrase", count(*) AS "Times"
FROM asfact_wc_searche_0 f
  JOIN asdim_time t ON f.timeid = t.id
  JOIN asdim_users u ON f.userid = u.id
  JOIN asdim_wc_searche_0 s ON f.searched_phrase_ = s.id
WHERE u.userid = 'user1'
  AND f.timeid >= (select id
                   from asdim_time
                   where year = 2011
                     and monthofyear = 0
                     and dayofmonth = 1
                     and hourofday = 0)
    AND f.timeid <  (select id
                   from asdim_time
                   where year = 2011
                     and monthofyear = 1
                     and dayofmonth = 1
                     and hourofday = 0)
GROUP BY u.userid, s.phrase_
ORDER BY count(*) DESC, s.phrase_
```

Sample Report Output

```
Search Phrase      Times
-----------------------
product prices         4
sales report 2011      1
```

47.3.4.6 Sample SQL: Page Views

The following query returns the pages viewed by a specific user during a specific time period.

```
SELECT u.userid AS "User", count(*) AS "Views"
FROM asfact_wc_pagevie_0 f
  JOIN asdim_time t ON f.timeid = t.id
  JOIN asdim_users u ON f.userid = u.id
  JOIN asdim_wc_pages_0 p ON f.page_ = p.id
  JOIN asdim_wc_groupsp_0 g ON f.groupspace_ = g.id
WHERE g.name_ = 'Space 1'
  AND p.name_ = 'Page 1'
  AND f.timeid >= (select id
                   from asdim_time
                   where year = 2011
                     and monthofyear = 0
                     and dayofmonth = 1
                     and hourofday = 0)
  AND f.timeid <  (select id
                   from asdim_time
                   where year = 2011
                     and monthofyear = 1
                     and dayofmonth = 1
                     and hourofday = 0)
```

```
GROUP BY u.userid
ORDER BY count(*) DESC
```

Sample Report Output

```
User    Views
--------------
user3     245
user1     190
user2      65
```

48 Integrating the Lists Service

This chapter provides an overview of the Lists service and describes how to integrate the Lists service with a Framework application. It includes the following sections:

- Section 48.1, "Introduction to the Lists Service"
- Section 48.2, "Basic Configuration for the Lists Service"
- Section 48.3, "Adding Lists to a Resource Catalog"
- Section 48.4, "Using the Lists Service REST APIs"
- Section 48.5, "Troubleshooting the Lists Service"

48.1 Introduction to the Lists Service

This section provides an overview of the Lists service features and requirements. It includes the following subsections:

- Section 48.1.1, "Understanding the Lists Service"
- Section 48.1.2, "What Happens to the Lists Service at Runtime"

48.1.1 Understanding the Lists Service

The Lists service provides a means of creating lists and exposing them for placement on application pages at runtime. At design time, you can make the Lists task flow available in your runtime Resource Catalog. At runtime, users can add the task flow from the Catalog to a page and use the task flow to create lists.

At design time, you can add a code snippet to the Resource Catalog to ensure that populated lists are added to the Catalog at runtime, making them available for placement on your application pages.

By default, only users assigned the seeded role `Administrator` have manage permission on lists. This means at runtime such users can create, edit, and delete lists and edit list data. The default permission can be changed at deployment in `jazn-data.xml` or by using Enterprise Manager or the Application Policy Manager. Authorized users can change this permission at runtime at the component level (for more information, see Chapter 24, "Modifying Default Security Behavior of Composer Components"). Only role-based permission is supported.

48.1.2 What Happens to the Lists Service at Runtime

At runtime, the Lists task flow provides controls for creating lists and adding list data. For information about these controls, see the sections, "Creating and Managing Lists,"

and "Adding and Managing List Data," in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

> **Tip:** To enable users to work with lists at runtime, they must be explicitly permitted to do so. For more information, see Section 48.2.3, "Enabling Users to Work with Lists at Runtime."

48.2 Basic Configuration for the Lists Service

This section describes the steps required for adding the Lists service to your application. It includes the following subsections:

- Section 48.2.1, "Setting Up Connections for the Lists Service"
- Section 48.2.2, "Adding the Lists Service at Design Time"
- Section 48.2.3, "Enabling Users to Work with Lists at Runtime"

48.2.1 Setting Up Connections for the Lists Service

The Lists service requires a connection to the database where the `WEBCENTER`is installed. The Lists service stores all list data in the database. For details about setting up a database connection to the database where the `WEBCENTER` is installed, see Section 7.2.2, "Setting Up a Database Connection."

> **Note:** For details about installing the database and the `WEBCENTER`, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

48.2.2 Adding the Lists Service at Design Time

This section explains a basic integration of the Lists service. It includes the following subsections:

- Section 48.2.2.1, "Lists Service Task Flows"
- Section 48.2.2.2, "How to Add the Lists Service to your Application"

48.2.2.1 Lists Service Task Flows

The Lists service includes one task flow: Lists (Figure 48–1).

Figure 48–1 A List Rendered in the Lists Task Flow



For information about the Lists task flow, see Section 48.1.2, "What Happens to the Lists Service at Runtime."

48.2.2.2 How to Add the Lists Service to your Application

To add the Lists task flow to your Framework application:

1. Follow the steps described in Section 7.2, "Preparing Your Framework Application to Consume Services." to implement security and create a new customizable page in your application.
2. Create a JSF page on which to add the Lists task flow.
3. Configure security on the page.

   See Also: For more information, see Chapter 24, "Modifying Default Security Behavior of Composer Components."

4. In the Resource Palette, open **My Catalogs**, then **WebCenter Portal - Services Catalog**, then the **Task Flows** folder.
5. Drag and drop the **Lists** task flow onto your page, and select **Region** from the context menu.
6. Save and run your page to the browser.

48.2.3 Enabling Users to Work with Lists at Runtime

Out of the box, only the application administrator can work with lists. For users to see and work with lists at runtime, they must explicitly be granted, minimally, the permission `ListPermission:view`. At runtime, application administrators can assign this permission through the Role Manager task flow.

For example, to grant view access to lists to all authenticated users, the administrator can create the role `AllUsers` in the Role Manager task flow, add `authenticated-role` as a member, and grant view access on Lists.

If you plan to add the Lists service to your application, then you must also add the Role Manager task flow. If your application is based on a WebCenter , then the Role Manager task flow is available out of the box on the **Security** tab in **Administration** pages. At runtime, you can navigate to this page using the following URL:

`http://`*`host:port/appcontxtroot`*`/faces/admin`

Note: Roles and permissions created and granted through the Role Manager at design time are not packaged in the application and are not available after deployment. The provisioning steps described in this section must be taken after deployment.

See Also: For information about the Role Manager task flow, see Section 69.4, "Using the Role Manager Task Flow."

48.3 Adding Lists to a Resource Catalog

If properly configured, the Resource Catalog makes populated lists available at runtime, enabling authorized users to add them to pages in edit mode. The procedure for adding components to an existing resource catalog or creating a custom catalog are described in Chapter 16, "Creating and Managing Resource Catalogs." This section provides an example of a code snippet to use to add lists to a resource catalog (Example 48–1).

Note: The values in uppercase are message keys for translated strings.

Example 48–1 Code to Add Lists to a Resource Catalog

```
<customFolder id="groupLists" visible="#{true}"
          factoryClass="oracle.webcenter.list.view.rc.ListServiceContextFactory">
     <attributes>
       <attribute value="LISTS_CUSTOM_FOLDER.TITLE" attributeId="Title"
          isKey="true"/>
       <attribute value="LISTS_CUSTOM_FOLDER.DESCRIPTION"
          attributeId="Description" isKey="true"/>
       <attribute value="LISTS_CUSTOM_FOLDER.KEYWORDS"
          attributeId="Subject" isKey="true"/>
       <attribute value="oracle.webcenter.list"
          attributeId="WEBCENTER_SERVICE_ID" isKey="false"/>
       <attribute value="/adf/webcenter/folderlists_qualifier.png"
          attributeId="IconURI"/>
     </attributes>
</customFolder>
```

48.4 Using the Lists Service REST APIs

Oracle WebCenter Portal provides REST APIs to allow access to Lists functionality through interfaces other than the provided task flows. You can use the Lists service REST APIs to perform the following actions:

- Work with lists in a space, such as retrieving a list of lists or adding a new list
- Work with one list, such as retrieving or updating list metadata, or deleting the list
- Work with rows in a list
- Work with a list row, such as retrieving, updating, or deleting the row
- Work with list columns, such as adding a column
- Work with a list column, such as retrieving, updating, or deleting the column

This section describes the associated with the Lists service. It includes the following subsections:

- Section 48.4.1, "Entry Point for Lists"
- Section 48.4.2, "Lists Resource Type Taxonomy"
- Section 48.4.3, "Lists Security Considerations"
- Section 48.4.4, "Lists Resource Types"

> **See Also:** For an introduction to the REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

48.4.1 Entry Point for Lists

To get to the REST entry point for lists in a space of the Spaces application, you must navigate to a specific space to see a link to lists in that space. For all the lists in a space, find the link elements with a `resourceType` of:

```
urn:oracle:webcenter:space:lists
```

The corresponding `href` or `template` element provides the URI entry point, which retrieves the lists in the space.

> **See Also:** For more information about the Resource Index, see Section 54.5.1, "The Resource Index."
>
> For more information about resource types, see Section 54.5.2.1, "Resource Type."

48.4.2 Lists Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. The resource type taxonomy for lists is:

```
urn:oracle:webcenter:space:lists
urn:oracle:webcenter:space:list
     urn:oracle:webcenter:space:list:rows
     urn:oracle:webcenter:space:list:row
     urn:oracle:webcenter:space:list:columns
     urn:oracle:webcenter:space:list:column
```

48.4.3 Lists Security Considerations

You must be logged in to the REST service to access any of the Lists REST APIs. After that, the underlying service handles permission checking.

> **See Also:** For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

48.4.4 Lists Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- Section 48.4.4.1, "urn:oracle:webcenter:space:lists"
- Section 48.4.4.2, "urn:oracle:webcenter:space:list"

- Section 48.4.4.3, "urn:oracle:webcenter:space:list:rows"
- Section 48.4.4.4, "urn:oracle:webcenter:space:list:row"
- Section 48.4.4.5, "urn:oracle:webcenter:space:list:columns"
- Section 48.4.4.6, "urn:oracle:webcenter:space:list:column"

48.4.4.1 urn:oracle:webcenter:space:lists

The `lists` response provides a means of retrieving the lists in a given space (`GET`) and adding lists to a space (`POST`). This section includes the following subsections:

- Section 48.4.4.1.1, "Navigation Paths to lists"
- Section 48.4.4.1.2, "Supported Methods for lists"
- Section 48.4.4.1.3, "Resource Types Linked to from lists"

48.4.4.1.1 Navigation Paths to lists This section shows how the client can navigate through the hypermedia to access the `lists` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
          urn:oracle:webcenter:space:resourceindex
               urn:oracle:webcenter:space:lists
```

48.4.4.1.2 Supported Methods for lists The `lists` resource type supports the following methods:

- Method (lists): `GET`
- Method (lists): `POST`

Method (lists): `GET`

This method retrieves the lists in a space.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists?q=name:equals:Pr
  ojectIssues&utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: none
- Request headers: [Accept = `application/xml | application/json`]
- Request parameters - `startIndex`, `itemsPerPage`, `q`, `projection`
  - `q` parameter for query, format `'q=attribute:operator:value'`, for example:

    ```
    'q=name:equals:ProjectIssues'
    ```

    * Supported operators for Strings: `equals`, `not.equals`, `contains`, `starts.with`
    * Supported operators for Dates: `equals`, `not.equals`, `greater.than`, `greater.than.or.equals`, `less.than`, `less.than.or.equals`
    * May specify several conditions separated by semicolon (;)
- Searchable attributes (Table 48–1)

Table 48–1 Searchable Attributes for lists GET Method

Element	Type	Description
name	string	Name of list
description	string	Description of list
creator	string	User who created list
created	date	Date the list was created
modifier	string	User who modified list
modified	date	Date the list was last modified

- Response status: `200` [OK]
- Response body: `<lists>`
- Example:

```
<lists resourceType="urn:oracle:webcenter:space:lists">
     <links>
          <link resourceType="urn:oracle:webcenter:list:lists"
               rel="self" href="opaque"/>
          <link template="opaque?startIndex={startIndex}&
               itemsPerPage={itemsPerPage}&q={searchTerms}&
               projection={projection}
               &stoken=FDgsOu7a2NTTM1cLJvnpkDXfihtHx5Q*"
               resourceType="urn:oracle:webcenter:list:lists"/>
     </links>
     <items>
          <list resourceType="urn:oracle:webcenter:space:list">
               <links>
                    <link resourceType="urn:oracle:webcenter:list" rel="self"
                         href="opaque"
                         capabilities="urn:oracle:webcenter:update
                         urn:oracle:webcenter:delete"/>
                    <link resourceType="urn:oracle:webcenter:list:rows"
                         href="opaque"
                         capabilities="urn:oracle:webcenter:create"/>
                    <link template="opaque?
                         startIndex={startIndex}&
                         itemsPerPage={itemsPerPage}&q={searchTerms}&
                         stoken=FDgsOu7a2NTTM1cLJvnpkDXfihtHx5Q*"
                         resourceType="urn:oracle:webcenter:list:rows"/>
                    <link resourceType="urn:oracle:webcenter:list:columns"
                         href="opaque"
                         capabilities="urn:oracle:webcenter:create"/>
               </links>
               <id>/oracle/webcenter/list/scopedMD/
                    se0dea180_e2c1_45ac_b08b_
                    ba2c0b26aa72/lists/ProjectIssues.xml</id>
               <name>Project Issues</name>
               <description/>
          </list>
     </items>
</lists>
```

Method (lists): `POST`

This method creates a list in a space.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists?utoken=FKld8lalI
  3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: `<list>`
- Writable elements (Table 48–2)

Table 48–2 Create List Writable Elements

Element	Type	Constraints	Description
name[1]	string	—	Name of list
description	string	—	Description of list
columns[1]	`urn:oracle:webcenter:space:list:columns`	1 to 30	List columns

[1] Denotes required element

- Example:

```
<list>
     <name>RestExample</name>
     <description>Created using rest api</description>
     <columns>
          <items>
               <metaColumn>
                    <name>No.</name>
                    <dataType>number</dataType>
               </metaColumn>
               <metaColumn>
                    <name>Description</name>
                    <dataType>string</dataType>
               </metaColumn>
          </items>
     </columns>
</list>
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `201` [Created]
- Response body: `<list>`
- Retrieved elements (Table 48–3)

Table 48–3 Retrieved Elements from lists

Element	Type	Description
id	string	List ID
name	string	List name
description	string	List description
scope.guid	string	GUID of space to which the list belongs
scope.name	string	Name of the space to which the list belongs
creator	string	Name of the user who created the list
created	date	Date the list was created

Table 48–3 (Cont.) Retrieved Elements from lists

Element	Type	Description
modifier	string	Name of the user who last modified the list
modified	date	Date the list was last modified
columns	urn:oracle:webcenter:space:list:columns	List columns

- Example:

```
<list resourceType="urn:oracle:webcenter:space:list">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list" rel="self"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/
               lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
          <link template="http://dadvmh0330.us.oracle.com:8889
               /rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
               s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
               417b_a35f_fb03874979c0.xml)/rows?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_
               w**&startIndex={startIndex}&itemsPerPage={itemsPerPage}
               &q={searchTerms}&utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               resourceType="urn:oracle:webcenter:space:list:rows"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
               vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)/rows?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
          <link resourceType="urn:oracle:webcenter:space:list:columns"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
               vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)/columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_
               w**" capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
     </links>
     <id>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
          f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml</id>
     <name>RestExample</name>
     <description>Created using rest api</description>
     <scope>
          <guid>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</guid>
          <name>vs1</name>
     </scope>
     <creator>weblogic</creator>
     <author>
          <links>
               <link resourceType="urn:oracle:webcenter:people:person"
                    rel="via" href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
                    @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="image/png"
                    template="http://dadvmh0330.us.oracle.com:8889/
```

```
                    webcenter/profilephoto/45394133354538304630424
                    331314446424642453046453333394535354232231/{size}?
                    _xResourceMethod=wsrp"
                    resourceType="urn:oracle:webcenter:people:person"
                    rel="urn:oracle:webcenter:people:icon"
                    href="http://dadvmh0330.us.oracle.com:8889/webcenter/
                    profilephoto/45394133354538304630424331314446424624530464533333394535423231/SMALL?_xResourceMethod=wsrp"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="text/html"
                    resourceType="urn:oracle:webcenter:spaces:profile"
                    rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
                    webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
                    peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
                    capabilities="urn:oracle:webcenter:read"/>
          </links>
          <displayName>weblogic</displayName>
          <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
          <id>weblogic</id>
     </author>
     <created>2010-11-18T06:10:32.250-08:00</created>
     <modifier>weblogic</modifier>
     <modifiedByUser>
          <displayName>weblogic</displayName>
          <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
          <id>weblogic</id>
     </modifiedByUser>
     <modified>2010-11-18T06:10:32.250-08:00</modified>
     <columns resourceType="urn:oracle:webcenter:space:list:columns">
          <links>
               <link resourceType="urn:oracle:webcenter:space:list:columns"
                    rel="self"
                    href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces
                    /vs1/lists/(/oracle/webcenter/list/scopedMD/
                    s355923f0_2f04_4fd0_83ad_
                    f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                    fb03874979c0.xml)/columns?utoken=
                    FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read
                    urn:oracle:webcenter:create"/>
               <link resourceType="urn:oracle:webcenter:space:list"
                    rel="urn:oracle:webcenter:parent"
                    href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
                    s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/
                    ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)?
                    utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"/>
               </links>
               <items>
                    <metaColumn resourceType=
                         "urn:oracle:webcenter:space:list:column">
                         <links>
                              <link resourceType="urn:oracle:webcenter:space:
                                   list:column" rel="self
                                   "href="http://dadvmh0330.us.oracle.com:8889/
                                   rest/api/spaces/vs1/lists/
                                   (/oracle/webcenter/list/scopedMD/
                                   s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/
                                   lists/ls_c9cecbc7_756b_417b_a35f_
                                   fb03874979c0.xml)/columns/
```

```
                                        (lco_9bdd1418_6004_40ba_a052_04e8335b7ee8)?
                                        utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                                        capabilities="urn:oracle:webcenter:read
                                        urn:oracle:webcenter:update
                                        urn:oracle:webcenter:delete"/>
                                   </links>
                                   <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
                                   <name>No.</name>
                                   <dataType>number</dataType>
                                   <required>false</required>
                                   <displayLength>10</displayLength>
                                   <format>number</format>
                                   <allowLinks>false</allowLinks>
                              </metaColumn>
                              <metaColumn resourceType=
                                   "urn:oracle:webcenter:space:list:column">
                                   <links>
                                        <link resourceType=
                                        "urn:oracle:webcenter:space:list:column"
                                        rel="self"
                                        href="http://dadvmh0330.us.oracle.com:8889/
                                        rest/api/spaces/vs1/lists/(/oracle/
                                        webcenter/list/scopedMD/
                                        s355923f0_2f04_4fd0_83ad_
                                        f7dac2a7ceed/lists/ls_c9cecbc7_756b_
                                        417b_a35f_fb03874979c0.xml)/columns/
                                        (lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24)?
                                        utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                                        capabilities="urn:oracle:webcenter:read
                                        urn:oracle:webcenter:update
                                        urn:oracle:webcenter:delete"/>
                              </links>
                              <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
                              <name>Description</name>
                              <dataType>string</dataType>
                              <required>false</required>
                              <maxLength>500</maxLength>
                              <displayLength>20</displayLength>
                              <allowLinks>false</allowLinks>
                              <editLines>1</editLines>
                         </metaColumn>
                    </items>
               </columns>
          </list>
```

48.4.4.1.3 Resource Types Linked to from lists Table 48–4 lists the resource types that the client can link to from the `lists` resource.

Table 48–4 Resource Types Linked to from lists

rel	resourceType
self	`urn:oracle:webcenter:space:lists`
	`urn:oracle:webcenter:space:list`

48.4.4.2 urn:oracle:webcenter:space:list

The `list` response provides a means of retrieving, updating, and deleting an individual list. This section includes the following subsections:

- Section 48.4.4.2.1, "Navigation Paths to list"
- Section 48.4.4.2.2, "Supported Methods for list"
- Section 48.4.4.2.3, "Resource Types Linked to from list"

48.4.4.2.1 Navigation Paths to list This section shows how the client can navigate through the hypermedia to access the `list` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
          urn:oracle:webcenter:space:resourceindex
               urn:oracle:webcenter:space:lists
                    urn:oracle:webcenter:space:list
```

48.4.4.2.2 Supported Methods for list The list resource type supports the following methods:

- Method (list): `GET`
- Method (list): `PUT`
- Method (list): `DELETE`

Method (list): `GET`

Use this method to retrieve a list.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_5fb62dc3514d.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: none
- Request headers: [Accept = `application/xml | application/json`]
- Response status: `200` [OK]
- Response body: `<list>`
- Retrieved elements (Table 48–5)

Table 48–5 Retrieved Elements for list

Element	Type	Description
id	string	ID of the list
name	string	Name of the list
description	string	Description of the list
scope.guid	string	GUID of the space to which the list belongs
scope.name	string	Name of the space to which the list belongs
creator	string	User who created the list
created	Date	Date on which the list was created
modifier	string	User who last modified the list
modified	Date	Date on which the list was last modified
columns	`urn:oracle:webcenter:space:list:columns`	The columns that make up the list

- For example:

```
<list resourceType="urn:oracle:webcenter:space:list">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list" rel="self"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/
               lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
               f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update
               urn:oracle:webcenter:delete"/>
          <link template="http://dadvmh0330.us.oracle.com:8889
               /rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
               s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
               417b_a35f_fb03874979c0.xml)/rows?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**&
               startIndex={startIndex}&itemsPerPage={itemsPerPage}&q=
               {searchTerms}&utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               resourceType="urn:oracle:webcenter:space:list:rows"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/
               spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_
               2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)/rows?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
          <link resourceType="urn:oracle:webcenter:space:list:columns"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
               vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)/columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_
               w**" capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
     </links>
     <id>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
          f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml</id>
     <name>RestExample</name>
     <description>Created using rest api</description>
     <scope>
          <guid>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</guid>
          <name>vs1</name>
     </scope>
     <creator>weblogic</creator>
     <author>
          <links>
               <link resourceType="urn:oracle:webcenter:people:person"
                    rel="via" href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
                    @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="image/png"
                    template="http://dadvmh0330.us.oracle.com:8889/
                    webcenter/profilephoto/45394133354538304630424331314444
                    64246424530464533333945353542323 1/{size}?
                    _xResourceMethod=wsrp"
                    resourceType="urn:oracle:webcenter:people:person"
                    rel="urn:oracle:webcenter:people:icon"
                    href="http://dadvmh0330.us.oracle.com:8889/
                    webcenter/profilephoto/45394133354538304630424331314444
                    64246424530464533333945353542323 1/SMALL?
                    _xResourceMethod=wsrp"
```

```
                capabilities="urn:oracle:webcenter:read"/>
            <link type="text/html"
                resourceType="urn:oracle:webcenter:spaces:profile"
                rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
                webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
                peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
                capabilities="urn:oracle:webcenter:read"/>
        </links>
        <displayName>weblogic</displayName>
        <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
        <id>weblogic</id>
    </author>
    <created>2010-11-18T06:10:32.250-08:00</created>
    <modifier>weblogic</modifier>
    <modifiedByUser>
        <displayName>weblogic</displayName>
        <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
        <id>weblogic</id>
    </modifiedByUser>
    <modified>2010-11-18T06:10:32.250-08:00</modified>
    <columns resourceType="urn:oracle:webcenter:space:list:columns">
        <links>
            <link resourceType="urn:oracle:webcenter:space:list:columns"
                rel="self" href="http://dadvmh0330.us.oracle.com:8889/
                rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
                s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_
                756b_417b_a35f_fb03874979c0.xml)/
                columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                capabilities="urn:oracle:webcenter:read
                urn:oracle:webcenter:create"/>
            <link resourceType="urn:oracle:webcenter:space:list"
                rel="urn:oracle:webcenter:parent"
                href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
                vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_
                4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                fb03874979c0.xml)?utoken=
                FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"/>
        </links>
        <items>
            <metaColumn resourceType=
                "urn:oracle:webcenter:space:list:column">
                <links>
                    <link resourceType=
                        "urn:oracle:webcenter:space:list:column"
                        rel="self"
                         href="http://dadvmh0330.us.oracle.com:8889/
                        rest/api/spaces/vs1/lists/(/oracle/webcenter/
                        list/scopedMD/s355923f0_2f04_4fd0_83ad_
                        f7dac2a7ceed/lists/ls_c9cecbc7_
                        756b_417b_a35f_fb03874979c0.xml)/columns/(lco_
                        9bdd1418_6004_40ba_a052_04e8335b7ee8)?
                        utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                        capabilities="urn:oracle:webcenter:read
                        urn:oracle:webcenter:update
                        urn:oracle:webcenter:delete"/>
                </links>
                <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
                <name>No.</name>
                <dataType>number</dataType>
                <required>false</required>
```

```
                    <displayLength>10</displayLength>
                    <format>number</format>
                    <allowLinks>false</allowLinks>
               </metaColumn>
               <metaColumn resourceType=
                    "urn:oracle:webcenter:space:list:column">
                    <links>
                         <link resourceType=
                              "urn:oracle:webcenter:space:list:column"
                              rel="self"
                              href="http://dadvmh0330.us.oracle.com:8889
                              /rest/api/spaces/vs1/lists/(/oracle/webcenter/
                              list/scopedMD/s355923f0_2f04_4fd0_83ad_
                              f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                              fb03874979c0.xml)/columns/(lco_3a31fd20_24f1_
                              4422_99fd_c00d7bed4b24)?
                              utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                              capabilities="urn:oracle:webcenter:read
                              urn:oracle:webcenter:update
                              urn:oracle:webcenter:delete"/>
                    </links>
                    <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
                    <name>Description</name>
                    <dataType>string</dataType>
                    <required>false</required>
                    <maxLength>500</maxLength>
                    <displayLength>20</displayLength>
                    <allowLinks>false</allowLinks>
                    <editLines>1</editLines>
               </metaColumn>
          </items>
     </columns>
</list>
```

Method (list): `PUT`

Use this method to update a list name or description.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
  er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_
  4d49_9354_5fb62dc3514d.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: `<list>`
- Writable elements for list (Table 48–6)

Table 48–6 Writable Elements for list

Element	Type	Required	Constraints	Description
name[1]	string	Yes	1 or more characters	Name of this list
description	string	No	none	Description of this list

[1] Denotes required element

- For example:

  ```
  <list>
       <name>RestExampleUpdate</name>
  ```

```
    <description>Updated using rest api</description>
</list>
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `200` [OK]
- Response body: `<list>`

Method (list): `DELETE`

Use this method to delete a list.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_5fb62dc3514d.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Response status: `204` [No Content]
- Response body: none

48.4.4.2.3 Resource Types Linked to from list Table 48–7 lists the resource types that the client can link to from the `list` resource.

Table 48–7 Resource Types Linked to from list

rel	resourceType
self	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:rows`
	`urn:oracle:webcenter:space:list:columns`

48.4.4.3 urn:oracle:webcenter:space:list:rows

The `rows` response provides a means of retrieving and adding rows to a list. This section includes the following subsections:

- Section 48.4.4.3.1, "Navigation Paths to rows"
- Section 48.4.4.3.2, "Supported Methods for rows"
- Section 48.4.4.3.3, "Resource Types Linked to from rows"

48.4.4.3.1 Navigation Paths to rows This section shows how the client can navigate through the hypermedia to access the `rows` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
          urn:oracle:webcenter:space:resourceindex
               urn:oracle:webcenter:space:lists
                    urn:oracle:webcenter:space:list
                         urn:oracle:webcenter:space:list:rows
```

48.4.4.3.2 Supported Methods for rows The rows resource type supports the following methods:

- Method (rows): `GET`

- Method (rows): POST

Method (rows): GET

Use this method to retrieve list rows.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_5fb62dc3514d.xml)/rows?q=lco_9bdd1418_6004_40ba_a052_04e8335b7ee8:equals:2&utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Request headers: [Accept = `application/xml | application/json`]
- Request parameters - `startIndex`, `itemsPerPage`, `q`, `projection`
  - `q` parameter for query, format `'q=columnId:operator:value'`, for example:

    ```
    'q=lco_9bdd1418_6004_40ba_a052_04e8335b7ee8:equals:2'
    ```

    * Supported operators for `Strings`: `equals`, `not.equals`, `contains`, `starts.with`
    * Supported operators for `Numbers`, `Dates`: `equals`, `not.equals`, `greater.than`, `greater.than.or.equals`, `less.than`, `less.than.or.equals`
    * May specify several conditions separated by semi-colon (;)
- Searchable attributes for rows: Rows are searchable by column values.
- Response status: `200` [OK]
- Response body: `<rows>`
- For example:

```
<rows resourceType="urn:oracle:webcenter:space:list:rows">
     <links>
          <link template="http://dadvmh0330.us.oracle.com:8889/
               rest/api/spaces/vs1/lists/(/oracle/webcenter/list/
               scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_
               c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/rows?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**&
               startIndex={startIndex}&itemsPerPage={itemsPerPage}&
               q={searchTerms}&utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               resourceType="urn:oracle:webcenter:space:list:rows" rel="self"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
               vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)/rows?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
          <link resourceType="urn:oracle:webcenter:space:list"
               rel="urn:oracle:webcenter:parent"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/
               vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"/>
     </links>
     <itemsPerPage>1</itemsPerPage>
```

```
<startIndex>0</startIndex>
<items>
     <row resourceType="urn:oracle:webcenter:space:list:row">
          <links>
               <link resourceType="urn:oracle:webcenter:space:list:row"
                    rel="self" href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/spaces/vs1/lists/(/oracle/webcenter/list/
                    scopedMD/s355923f0_2f04_4fd0_83ad_
                    f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                    fb03874979c0.xml)/rows/(lr_a14d427f_8515_4c82_956f_
                    a60e6e08668c)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read
                    urn:oracle:webcenter:update
                    urn:oracle:webcenter:delete"/>
          </links>
          <id>lr_a14d427f_8515_4c82_956f_a60e6e08668c</id>
          <listId>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_
               83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml</listId>
          <scope>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</scope>
          <creator>weblogic</creator>
          <author>
               <links>
                    <link resourceType=
                         "urn:oracle:webcenter:people:person"
                         rel="via"
                         href="http://dadvmh0330.us.oracle.com:8889/
                         rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/
                         lists/@self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_
                         w**" capabilities="urn:oracle:webcenter:read"/>
                    <link type="image/png"
                         template="http://dadvmh0330.us.oracle.com:8889/
                         webcenter/profilephoto/4539413335453830463042 4
                         331314446424642453046453333394535423231/
                         {size}?_xResourceMethod=wsrp"
                         resourceType="urn:oracle:webcenter:people:person"
                         rel="urn:oracle:webcenter:people:icon"
                         href="http://dadvmh0330.us.oracle.com:8889/
                         webcenter/profilephoto/4539413335453830463042 4
                         331314446424642453046453333394535423231/SMALL?
                         _xResourceMethod=wsrp"
                         capabilities="urn:oracle:webcenter:read"/>
                    <link type="text/html"
                         resourceType="urn:oracle:webcenter:spaces:
                         profile" rel="alternate"
                         href="http://dadvmh0330.us.oracle.com:8889/
                         webcenter/faces/oracle/webcenter/webcenterapp/
                         view/pages/peopleconn/UserProfileGallery.jspx?
                         wc.username=weblogic"
                         capabilities="urn:oracle:webcenter:read"/>
               </links>
               <displayName>weblogic</displayName>
               <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
               <id>weblogic</id>
          </author>
          <created>2010-11-18T07:12:06.599-08:00</created>
          <modifier>weblogic</modifier>
          <modifiedByUser>
               <links>
                    <link resourceType=
```

```
                                        "urn:oracle:webcenter:people:person" rel="via"
                                        href="http://dadvmh0330.us.oracle.com:8889/rest/
                                        api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists
                                        /@self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                                        capabilities="urn:oracle:webcenter:read"/>
                                   <link type="image/png"
                                        template="http://dadvmh0330.us.oracle.com:8889/
                                        webcenter/profilephoto/4539413335453830463042 4
                                        331314446424642453046453333394535535423231/
                                        {size}?_xResourceMethod=wsrp"
                                        resourceType="urn:oracle:webcenter:people:person"
                                        rel="urn:oracle:webcenter:people:icon"
                                        href="http://dadvmh0330.us.oracle.com:8889/
                                        webcenter/profilephoto/4539413335453830463042 4
                                        331314446424642453046453333394535535423231/SMALL?
                                        _xResourceMethod=wsrp"
                                        capabilities="urn:oracle:webcenter:read"/>
                                   <link type="text/html"
                                        resourceType="urn:oracle:webcenter:spaces:
                                        profile" rel="alternate"
                                        href="http://dadvmh0330.us.oracle.com:8889/
                                        webcenter/faces/oracle/webcenter/webcenterapp/
                                        view/pages/peopleconn/UserProfileGallery.jspx?
                                        wc.username=weblogic"
                                        capabilities="urn:oracle:webcenter:read"/>
                              </links>
                              <displayName>weblogic</displayName>
                              <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
                              <id>weblogic</id>
                         </modifiedByUser>
                         <modified>2010-11-18T07:12:06.599-08:00</modified>
                         <columns>
                              <column>
                                   <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
                                   <name>No.</name>
                                   <value>1.0</value>
                              </column>
                              <column>
                                   <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
                                   <name>Description</name>
                                   <value>test</value>
                              </column>
                         </columns>
                    </row>
               </items>
          </rows>
```

Method (rows): `POST`

Use this method to create a row in a list.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
  er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_
  4d49_9354_5fb62dc3514d.xml)/rows?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: `<row>`
- Writable elements (Table 48–8)

Table 48–8 Writable Elements for rows

Element	Type	Constraints	Description
columns.column.id[1]	string	—	Column ID
columns.column.value[1]	string	Valid value for column data type	Column value

[1] Denotes required element

- For example:

```
<row>
     <columns>
          <column>
               <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
               <value>1</value>
          </column>
          <column>
               <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
               <value>test</value>
          </column>
     </columns>
</row>
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `201` [Created]
- Response body: `<row>`
- Retrieved elements (Table 48–9)

Table 48–9 Retrieved Elements for rows

Element	Type	Description
id	string	Row ID
list id	string	List ID
scope	string	GUID of space to which list belongs
creator	string	Person who created the list
created	date	Date the list was created
modifier	string	Last user to modify the list
modified	date	Date the list was last modified
columns	—	Column values

- For example:

```
<row resourceType="urn:oracle:webcenter:space:list:row">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list:row" rel="self"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/
               lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
               f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_
               5fb62dc3514d.xml)/rows/(lr_4cc07327_49cb_4cd5_a270_
               182ddcc8db4a)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
```

```
</links>
<id>lr_4cc07327_49cb_4cd5_a270_182ddcc8db4a</id>
<listId>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
     f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_
     5fb62dc3514d.xml</listId>
<scope>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</scope>
<creator>weblogic</creator>
<author>
     <links>
          <link resourceType="urn:oracle:webcenter:people:person"
               rel="via" href="http://dadvmh0330.us.oracle.com:8889
               /rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
               @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read"/>
          <link type="image/png"
               template="http://dadvmh0330.us.oracle.com:8889/
               webcenter/profilephoto/45394133354538304630423431314446424624530
               4446424642453046453333394535354232 31/{size}?
               _xResourceMethod=wsrp"
               resourceType="urn:oracle:webcenter:people:person"
               rel="urn:oracle:webcenter:people:icon"
               href="http://dadvmh0330.us.oracle.com:8889/webcenter/
               profilephoto/45394133354538304630424331314446424624530
               4645333339453535423231/SMALL?_xResourceMethod=wsrp"
               capabilities="urn:oracle:webcenter:read"/>
          <link type="text/html"
               resourceType="urn:oracle:webcenter:spaces:profile"
               rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
               webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
               peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
               capabilities="urn:oracle:webcenter:read"/>
     </links>
     <displayName>weblogic</displayName>
     <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
     <id>weblogic</id>
</author>
<created>2010-11-17T06:34:25.042-08:00</created>
<modifier>weblogic</modifier>
<modifiedByUser>
     <links>
          <link resourceType="urn:oracle:webcenter:people:person"
               rel="via" href="http://dadvmh0330.us.oracle.com:8889/
               rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
               @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read"/>
          <link type="image/png"
               template="http://dadvmh0330.us.oracle.com:8889/
               webcenter/profilephoto/45394133354538304630423431314446424624530
               4446424642453046453333394535354232 31/{size}?
               _xResourceMethod=wsrp"
               resourceType="urn:oracle:webcenter:people:person"
               rel="urn:oracle:webcenter:people:icon"
               href="http://dadvmh0330.us.oracle.com:8889/webcenter/
               profilephoto/45394133354538304630424331314446424624530
               4645333339453535423231/SMALL?_xResourceMethod=wsrp"
               capabilities="urn:oracle:webcenter:read"/>
          <link type="text/html"
               resourceType="urn:oracle:webcenter:spaces:profile"
               rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
               webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
```

```
                    peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
                    capabilities="urn:oracle:webcenter:read"/>
            </links>
            <displayName>weblogic</displayName>
            <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
            <id>weblogic</id>
        </modifiedByUser>
            <modified>2010-11-17T06:34:25.042-08:00</modified>
        <columns>
            <column>
                <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
                <name>No.</name>
                <value>1.0</value>
            </column>
            <column>
                <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
                <name>Description</name>
                <value>test</value>
            </column>
        </columns>
    </row>
```

48.4.4.3.3 Resource Types Linked to from rows Table 48–10 lists the resource types that the client can link to from the `rows` resource.

Table 48–10 Resource Types Linked to from rows

rel	resourceType
self	`urn:oracle:webcenter:space:list.rows`
parent	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:row`

48.4.4.4 urn:oracle:webcenter:space:list:row

The `row` response provides a means of retrieving and adding, and deleting an individual list row. This section includes the following subsections:

- Section 48.4.4.4.1, "Navigation Paths to row"
- Section 48.4.4.4.2, "Supported Methods for row"
- Section 48.4.4.4.3, "Resource Types Linked to from row"

48.4.4.4.1 Navigation Paths to row This section shows how the client can navigate through the hypermedia to access the `row` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
          urn:oracle:webcenter:space:resourceindex
               urn:oracle:webcenter:space:lists
                    urn:oracle:webcenter:space:list
                         urn:oracle:webcenter:space:list:rows
                              urn:oracle:webcenter:space:list:row
```

48.4.4.4.2 Supported Methods for row The row resource type supports the following methods:

- Method (row): `GET`

- Method (row): PUT
- Method (row): DELETE

Method (row): GET

Use this method to retrieve data from a list row.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/rows/(lr_a14d427f_8515_4c82_956f_a60e6e08668c)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Request headers: [Accept = application/xml | application/json]
- Response status: 200 [OK]
- Response body: <row>
- Retrieved elements (Table 48–11)

Table 48–11 Retrieved Elements for row

Element	Type	Description
id	string	ID of the row
listId	string	ID of the parent list
scope	string	GUID of the space to which the list belongs
creator	string	User who created the row
created	Date	Date on which the row was created
modifier	string	User who last modified the row
modified	Date	Date on which the row was last modified
columns	urn:oracle:webcenter:space:list:columns	Column values

- For example:

```
<row resourceType="urn:oracle:webcenter:space:list:row">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list:row"
               rel="self" href="http://dadvmh0330.us.oracle.com:8889/
               rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD
               /s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_bb806754_0652_
               4d49_9354_5fb62dc3514d.xml)/rows/(lr_4cc07327_49cb_4cd5_a270_
               182ddcc8db4a)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
     </links>
     <id>lr_4cc07327_49cb_4cd5_a270_182ddcc8db4a</id>
     <listId>/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
          f7dac2a7ceed/lists/ls_bb806754_0652_4d49_9354_
          5fb62dc3514d.xml</listId>
     <scope>s355923f0_2f04_4fd0_83ad_f7dac2a7ceed</scope>
     <creator>weblogic</creator>
     <author>
          <links>
               <link resourceType="urn:oracle:webcenter:people:person"
```

```
                    rel="via" href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
                    @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="image/png"
                    template="http://dadvmh0330.us.oracle.com:8889/webcenter/
                    profilephoto/45394133354538304630424331314446424642453
                    464533339453535423231/{size}?_xResourceMethod=wsrp"
                    resourceType="urn:oracle:webcenter:people:person"
                    rel="urn:oracle:webcenter:people:icon"
                    href="http://dadvmh0330.us.oracle.com:8889/webcenter/
                    profilephoto/45394133354538304630424331314446424642453
                    464533339453535423231/SMALL?_xResourceMethod=wsrp"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="text/html"
                    resourceType="urn:oracle:webcenter:spaces:profile"
                    rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
                    webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
                    peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
                    capabilities="urn:oracle:webcenter:read"/>
          </links>
          <displayName>weblogic</displayName>
          <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
          <id>weblogic</id>
     </author>
     <created>2010-11-17T06:34:25.042-08:00</created>
     <modifier>weblogic</modifier>
     <modifiedByUser>
          <links>
               <link resourceType="urn:oracle:webcenter:people:person"
                    rel="via" href="http://dadvmh0330.us.oracle.com:8889/
                    rest/api/people/E9A35E80F0BC11DFBFBE0FE339E55B21/lists/
                    @self?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="image/png"
                    template="http://dadvmh0330.us.oracle.com:8889/webcenter/
                    profilephoto/45394133354538304630424331314446424642453
                    464533339453535423231/{size}?_xResourceMethod=wsrp"
                    resourceType="urn:oracle:webcenter:people:person"
                    rel="urn:oracle:webcenter:people:icon"
                    href="http://dadvmh0330.us.oracle.com:8889/webcenter/
                    profilephoto/45394133354538304630424331314446424642453
                    464533339453535423231/SMALL?_xResourceMethod=wsrp"
                    capabilities="urn:oracle:webcenter:read"/>
               <link type="text/html"
                    resourceType="urn:oracle:webcenter:spaces:profile"
                    rel="alternate" href="http://dadvmh0330.us.oracle.com:8889/
                    webcenter/faces/oracle/webcenter/webcenterapp/view/pages/
                    peopleconn/UserProfileGallery.jspx?wc.username=weblogic"
                    capabilities="urn:oracle:webcenter:read"/>
          </links>
          <displayName>weblogic</displayName>
          <guid>E9A35E80F0BC11DFBFBE0FE339E55B21</guid>
          <id>weblogic</id>
     </modifiedByUser>
     <modified>2010-11-17T06:34:25.042-08:00</modified>
     <columns>
          <column>
               <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
               <name>No.</name>
```

```
                <value>1.0</value>
            </column>
            <column>
                <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
                <name>Description</name>
                <value>test</value>
            </column>
        </columns>
</row>
```

Method (row): `PUT`

Use this method to update row data.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
417b_a35f_fb03874979c0.xml)/rows/(lr_a14d427f_8515_4c82_956f_
a60e6e08668c)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: `<row>`
- Writable elements (Table 48–12)

Table 48–12 Writable Elements for row

Element	Type	Constraints	Description
columns.column.id[1]	string	—	Column ID
columns.column.value[1]	string	valid value for column data type	Column value

[1] Denotes required element

- For example:

```
<row>
    <columns>
        <column>
            <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
            <value>1</value>
        </column>
        <column>
            <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
            <value>test</value>
        </column>
    </columns>
</row>
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `200` [OK]
- Response body: `<row>`

Method (row): `DELETE`

Use this method to delete a list row.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
```

```
er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
417b_a35f_fb03874979c0.xml)/rows/(lr_a14d427f_8515_4c82_956f_
a60e6e08668c)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Response status: `204` [No Content]
- Response body: none

48.4.4.4.3 Resource Types Linked to from row Table 48–13 lists the resource types that the client can link to from the `row` resource.

Table 48–13 Resource Types Linked to from row

rel	resourceType
self	`urn:oracle:webcenter:space:list:row`

48.4.4.5 urn:oracle:webcenter:space:list:columns

The `columns` response provides a means of retrieving and adding list columns. This section includes the following subsections:

- Section 48.4.4.5.1, "Navigation Paths to columns"
- Section 48.4.4.5.2, "Supported Methods for columns"
- Section 48.4.4.5.3, "Resource Types Linked to from columns"

48.4.4.5.1 Navigation Paths to columns This section shows how the client can navigate through the hypermedia to access the `columns` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
          urn:oracle:webcenter:space:resourceindex
               urn:oracle:webcenter:space:lists
                    urn:oracle:webcenter:space:list
                         urn:oracle:webcenter:space:list:columns
```

48.4.4.5.2 Supported Methods for columns The `columns` resource type supports the following methods:

- Method (columns): `GET`
- Method (columns): `POST`

Method (columns): `GET`

Use this method to retrieve columns in a list.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
417b_a35f_fb03874979c0.xml)/columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Request headers: [Accept = `application/xml | application/json`]
- Request parameters: none
- Response status: `200` [OK]

- Response body: `<metaColumns>`
- For example:

```
<metaColumns resourceType="urn:oracle:webcenter:space:list:columns">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list:columns"
               rel="self" href="http://dadvmh0330.us.oracle.com:8889/
               rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
               s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
               417b_a35f_fb03874979c0.xml)/columns?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:create"/>
          <link resourceType="urn:oracle:webcenter:space:list"
               rel="urn:oracle:webcenter:parent"
               href="http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/
               lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_
               f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
               fb03874979c0.xml)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"/>
     </links>
     <items>
          <metaColumn resourceType="urn:oracle:webcenter:space:list:column">
               <links>
                    <link resourceType="urn:oracle:webcenter:space:list:column"
                         rel="self" href="http://dadvmh0330.us.oracle.com:8889/
                         rest/api/spaces/vs1/lists/(/oracle/webcenter/list/
                         scopedMD/s355923f0_2f04_4fd0_83ad_
                         f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                         fb03874979c0.xml)/columns/(lco_9bdd1418_6004_40ba_
                         a052_04e8335b7ee8)?
                         utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                         capabilities="urn:oracle:webcenter:read
                         urn:oracle:webcenter:update
                         urn:oracle:webcenter:delete"/>
               </links>
               <id>lco_9bdd1418_6004_40ba_a052_04e8335b7ee8</id>
               <name>No.</name>
               <dataType>number</dataType>
               <required>false</required>
               <displayLength>10</displayLength>
               <format>number</format>
               <allowLinks>false</allowLinks>
          </metaColumn>
          <metaColumn resourceType="urn:oracle:webcenter:space:list:column">
               <links>
                    <link resourceType="urn:oracle:webcenter:space:list:column"
                         rel="self" href="http://dadvmh0330.us.oracle.com:8889/
                         rest/api/spaces/vs1/lists/(/oracle/webcenter/list/
                         scopedMD/s355923f0_2f04_4fd0_83ad_
                         f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_
                         fb03874979c0.xml)/columns/(lco_3a31fd20_24f1_4422_
                         99fd_c00d7bed4b24)?
                         utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
                         capabilities="urn:oracle:webcenter:read
                         urn:oracle:webcenter:update
                         urn:oracle:webcenter:delete"/>
               </links>
               <id>lco_3a31fd20_24f1_4422_99fd_c00d7bed4b24</id>
               <name>Description</name>
               <dataType>string</dataType>
```

```
                    <required>false</required>
                    <maxLength>500</maxLength>
                    <displayLength>20</displayLength>
                    <allowLinks>false</allowLinks>
                    <editLines>1</editLines>
                </metaColumn>
            </items>
        </metaColumns>
```

Method (columns): `POST`

Use this method to create a column in a list.

- Resource URI, for example:

  ```
  http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/columns?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
  ```

- Request body: `<metaColumn>`
- Writable elements (Table 48–14)

Table 48–14 Writable Elements for columns

Element	Type	Constraints	Description
name[1]	string	—	The name of the column
dataType[1]	string	▪ `string` ▪ `number` ▪ `datetime` ▪ `boolean` ▪ `person` ▪ `image` ▪ `richtext`	The data type of the column
required	boolean	▪ `true` ▪ `false`	Whether a value is required for the column
defaultValue	data type of column	Object must match data type	The default value of the column
maxLength	int	Valid only for `string` data type	The maximum length for a string value
rangeLow	int	Valid only for `number` data type	The low range value for a `number` data type
rangeHigh	int	Valid only for `number` data type	The high range value for a `number` data type
format	string	For `number` data type: ▪ number ▪ currency ▪ percent For `datetime` data type: ▪ date ▪ time ▪ both	The format of the column

Table 48–14 (Cont.) Writable Elements for columns

Element	Type	Constraints	Description
allowLinks	boolean	▪ `true` ▪ `false` Valid only for `string` data type	Whether a hyperlink can be specified for a column value
linkTarget	string	▪ `_blank` ▪ `_self` Valid only if `allowLinks=true`	`_blank` opens link in a new window, `_self` opens link in the same window
editLines	int	Valid only for `string` data type	The number of lines when editing a column value (default=1)
peopleScope	string	▪ GLOBAL ▪ SUB_SCOPE Valid only for `person` data type	Whether valid users are all users in directory or members of the space that contains the list
displayWidth	int	—	Display width of column in pixels
hint	string	—	Hint displayed to help user when entering column value

[1] Denotes required element

- For example:

```
<metaColumn>
     <name>Notes</name>
     <dataType>richtext</dataType>
</metaColumn
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `201` [Created]
- Response body: `<metaColumn>`
- Retrieved elements (Table 48–15)

Table 48–15 Retrieved Elements from columns

Element	Type	Description
id	string	Column ID
name	string	The name of the column
dataType	string	The data type of the column
required	boolean	Whether a value is required for the column
defaultValue	data type of column	The default value of the column
maxLength	int	The maximum length for a string value
rangeLow	int	The low range value for a `number` data type
rangeHigh	int	The high range value for a `number` data type
format	string	The format of the column
allowLinks	boolean	Whether a hyperlink can be specified for a column value
linkTarget	string	`_blank` opens link in a new window, `_self` opens link in the same window

Table 48–15 (Cont.) Retrieved Elements from columns

Element	Type	Description
editLines	int	The number of lines when editing a column value (default=1)
peopleScope	string	Whether valid users are all users in directory or members of the space that contains the list
displayWidth	int	Display width of column in pixels
hint	string	Hint displayed to help user when entering column value

- For example:

```
<metaColumn resourceType="urn:oracle:webcenter:space:list:column">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list:column"
               rel="self" href="http://dadvmh0330.us.oracle.com:8889/
               rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
               s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/
               ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/columns/
               (lco_fe2b9856_32f3_449a_a277_18dc7f6a779e)?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
     </links>
     <id>lco_fe2b9856_32f3_449a_a277_18dc7f6a779e</id>
     <name>Notes</name>
     <dataType>richtext</dataType>
     <required>false</required>
     <displayLength>20</displayLength>
     <allowLinks>false</allowLinks>
</metaColumn>
```

48.4.4.5.3 Resource Types Linked to from columns Table 48–16 lists the resource types that the client can link to from the `columns` resource.

Table 48–16 Resource Types Linked to from columns

rel	resourceType
self	`urn:oracle:webcenter:space:list.columns`
parent	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:column`

48.4.4.6 urn:oracle:webcenter:space:list:column

The `column` response provides a means of retrieving, adding, and deleting an individual list column. This section includes the following subsections:

- Section 48.4.4.6.1, "Navigation Paths to column"
- Section 48.4.4.6.2, "Supported Methods for column"
- Section 48.4.4.6.3, "Resource Types Linked to from column"

48.4.4.6.1 Navigation Paths to column This section shows how the client can navigate through the hypermedia to access the `column` resource:

```
urn:oracle:webcenter:resourceindex
     urn:oracle:webcenter:spaces
```

```
urn:oracle:webcenter:space:resourceindex
     urn:oracle:webcenter:space:lists
          urn:oracle:webcenter:space:list
               urn:oracle:webcenter:space:list:columns
                    urn:oracle:webcenter:space:list:column
```

48.4.4.6.2 Supported Methods for column The column resource type supports the following methods:

- Method (column): `GET`
- Method (column): `PUT`
- Method (column): `DELETE`

Method (column): `GET`

Use this method to retrieve a list column.

- Resource URI. for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/columns/(lco_fe2b9856_32f3_449a_a277_18dc7f6a779e)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Request headers: [Accept = `application/xml | application/json`]
- Response status: `200` [OK]
- Response body: `<metaColumn>`
- Retrieved elements (Table 48–17)

Table 48–17 column Retrieved Elements

Element	Type	Description
id	string	Column ID
name	string	The name of the column
dataType	string	The data type of the column
required	boolean	Whether a value is required for the column
defaultValue	data type of column	The default value of the column
maxLength	int	The maximum length for a string value
rangeLow	int	The low range value for a `number` data type
rangeHigh	int	The high range value for a `number` data type
format	string	The format of the column
allowLinks	boolean	Whether a hyperlink can be specified for a column value
linkTarget	string	`_blank` opens link in a new window, `_self` opens link in the same window
editLines	int	The number of lines when editing a column value (default=1)

Table 48–17 (Cont.) column Retrieved Elements

Element	Type	Description
peopleScope	string	Whether valid users are all users in directory or members of the space that contains the list
displayWidth	int	Display width of column in pixels
hint	string	Hint displayed to help user when entering column value

- For example:

```
<metaColumn resourceType="urn:oracle:webcenter:space:list:column">
     <links>
          <link resourceType="urn:oracle:webcenter:space:list:column"
               rel="self" href="http://dadvmh0330.us.oracle.com:8889/
               rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/
               s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/
               ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/
               columns/(lco_fe2b9856_32f3_449a_a277_18dc7f6a779e)?
               utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**"
               capabilities="urn:oracle:webcenter:read
               urn:oracle:webcenter:update urn:oracle:webcenter:delete"/>
     </links>
     <id>lco_fe2b9856_32f3_449a_a277_18dc7f6a779e</id>
     <name>Notes</name>
     <dataType>richtext</dataType>
     <required>false</required>
     <displayLength>20</displayLength>
     <allowLinks>false</allowLinks>
</metaColumn>
```

Method (column): `PUT`

Use this method to update column data.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcent
er/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_
417b_a35f_fb03874979c0.xml)/columns/(lco_fe2b9856_32f3_449a_a277_
18dc7f6a779e)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: `<metacolumn>`
- Writable elements (Table 48–18)

Table 48–18 column Writable Elements

Element	Type	Constraints	Description
name[1]	string	—	The name of the column
dataType[1]	string	▪ `string` ▪ `number` ▪ `datetime` ▪ `boolean` ▪ `person` ▪ `image` ▪ `richtext`	The data type of the column
required	boolean	▪ `true` ▪ `false`	Whether a value is required for the column
defaultValue	data type of column	Object must match data type	The default value of the column
maxLength	int	Valid only for `string` data type	The maximum length for a string value
rangeLow	int	Valid only for `number` data type	The low range value for a `number` data type
rangeHigh	int	Valid only for `number` data type	The high range value for a `number` data type
format	string	For `number` data type: ▪ number ▪ currency ▪ percent For `datetime` data type: ▪ date ▪ time ▪ both	The format of the column
allowLinks	boolean	▪ `true` ▪ `false` Valid only for `string` data type	Whether a hyperlink can be specified for a column value
linkTarget	string	▪ `_blank` ▪ `_self` Valid only if `allowLinks=true`	`_blank` opens link in a new window, `_self` opens link in the same window
editLines	int	Valid only for `string` data type	The number of lines when editing a column value (default=1)
peopleScope	string	▪ `GLOBAL` ▪ `SUB_SCOPE` Valid only for `person` data type	Whether valid users are all users in directory or members of the space that contains the list
displayWidth	int	—	Display width of column in pixels
hint	string	—	Hint displayed to help user when entering column value

[1] Denotes required element

- For example:

```
<metaColumn>
     <name>Comments</name>
     <dataType>richtext</dataType>
</metaColumn>
```

- Request headers: `Content-Type = application/xml | application/json`, [Accept = `application/xml | application/json`]
- Response status: `200` [OK]
- Response body: `<metaColumn>`

Method (column): `DELETE`

Use this method to delete a list column.

- Resource URI, for example:

```
http://dadvmh0330.us.oracle.com:8889/rest/api/spaces/vs1/lists/(/oracle/webcenter/list/scopedMD/s355923f0_2f04_4fd0_83ad_f7dac2a7ceed/lists/ls_c9cecbc7_756b_417b_a35f_fb03874979c0.xml)/columns/(lco_fe2b9856_32f3_449a_a277_18dc7f6a779e)?utoken=FKld8lalI3QRdi8TgQkOCGEzxL5x_w**
```

- Request body: none
- Response status: `204` [No Content]
- Response body: none

48.4.4.6.3 Resource Types Linked to from column Table 48–19 lists the resource types that the client can link to from the `column` resource.

Table 48–19 Resource Types Linked to from column

rel	resourceType
self	`urn:oracle:webcenter:space:list:column`

48.5 Troubleshooting the Lists Service

Problem

The List task flow shows an error that the WebCenter Portal's repository is not available.

Solution

The database connection is likely not configured. For more information, see Section 7.2.2, "Setting Up a Database Connection."

Problem

A 401 Unauthorized error appears.

Solution

It is likely that security is not set up for the application. Or it may be that there was insufficient permission to perform the attempted task.

49 Integrating Notifications

Notifications provides a means of subscribing to services and application objects and, when those objects change, receiving notification across one or more selected messaging channels. This chapter describes how to integrate Notifications with your Framework applications. It includes the following sections:

- Section 49.1, "Introduction to Notifications"
- Section 49.2, "Basic Configuration for Notifications"
- Section 49.3, "Advanced Information for Notifications"

49.1 Introduction to Notifications

This section provides an overview of Notifications. It includes the following subsections:

- Section 49.1.1, "Understanding Notifications"
- Section 49.1.2, "Requirements for Notifications"
- Section 49.1.3, "What Happens with Notifications at Runtime?"

49.1.1 Understanding Notifications

Notifications provides an automated means of triggering notices about subscribed services and objects across different messaging channels. Messages are triggered when the application services and objects to which a user has subscribed change. For example, a user can subscribe to a document and receive notification when another user changes the document.

Messaging channels can include phone text (SMS), mail, or Worklist, depending on what messaging servers are available and how the application administrator configures Notifications. For example, users can receive a mail message when a particular document changes, a text message when someone responds to a particular discussion topic, a Worklist alert when the user receives an invitation to connect. All messages contain links that take the user to the scene of the change.

Participating services and objects include People Connections (Connections, Message Board, and Feedback), Discussions, and Documents (including Wikis and Blogs).

Table 49–1 describes the types of activities that can trigger a notification and indicates the level at which the subscription is made.

Table 49–1 Activities that Can Trigger Notifications

Activity	Level
A user sends you an invitation to connect	Application
A user posts a message to your Message Board	Application
A user likes your Message Board post (messages explicitly set on a Message Board and not those added from Publisher to the Activity Stream)	Application
A user comments on your Message Board post (messages explicitly set on a Message Board and not those added from Publisher to the Activity Stream)	Application
A user posts feedback for you	Application
A user replies to a discussion topic	Object
A user comments on a discussion topic	Object
A user deletes a discussion topic	Object
A user comments on a document	Object
A user likes a document	Object
A user updates a document	Object
A user deletes a document	Object
A user comments on a wiki document	Object
A user likes a wiki document	Object
A user updates a wiki document	Object
A user deletes a wiki document	Object
A user comments on a blog entry	Object
A user likes a blog entry	Object
A user updates a blog entry	Object
A user deletes a blog entry	Object

Notifications leverages message delivery platforms available in WebCenter, such as the User Messaging Service (UMS) or the Mail service, to deliver the messages it generates. The Mail service furnishes the default messaging delivery mechanism, though you can instead use the BPEL connection provided through the Worklist service to widen a user's choice of channels to phone text, mail, and the Worklist.

You can set defaults for application-level subscriptions for all users through an XML configuration file (`notification-service-settings.xml`). You create your own version of this default file, and then use it to supersede the file of the same name that is provided for this purpose out of the box.

49.1.2 Requirements for Notifications

Before your application can work with the Notifications service, you must specify the Notification namespace in the application's `adf-config.xml` file. For more information, see Section 49.2.1, "Adding the Notifications Namespace."

Subscription capability is built into the services that support it and do not require additional configuration.

Notifications messaging capability relies on the presence of a configured Mail or Worklist connection to enable delivery of notifications messages. For Notifications messaging, you must explicitly configure the application to use either the BPEL server

that is configured for the Worklist service or the Mail server that is configured for the Mail service. For more information, see Section 49.2.2, "Creating a Notifications Connection to a Mail Server," and Section 49.2.3, "Creating a Notifications Connection to a BPEL Server."

To set application defaults for all users for application-level subscription preferences, you must provide your own version of the default file `notification-service-settings.xml`. For more information, see Section 49.2.4, "Setting Application Defaults for Notifications."

For Notifications to function properly, its task flows must reside on a secured page. This is because the current user must be known through authentication in order to track and report on subscribed actions. If the page or application that contains Notifications is not secured, you will encounter exception errors. For more information, see Section 49.2.6, "Setting Security for Notifications."

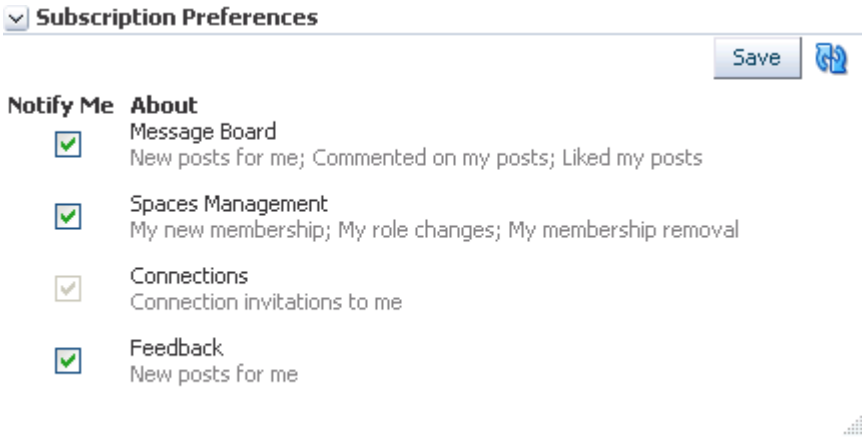
49.1.3 What Happens with Notifications at Runtime?

At runtime, users set up their preferred messaging channels through UMS Preferences. For more information, see the section "Establishing and Managing Your Messaging Channels and Filters" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

> **Note:** In designing your application, you must provide users with a way to access UMS preferences. For example, you can add a link to them in your UI, or you can point users to the Worklist task flow, which includes a link to them.

Users establish their application-level subscriptions through the Notifications task flow *Subscription Preferences* (Figure 49–1).

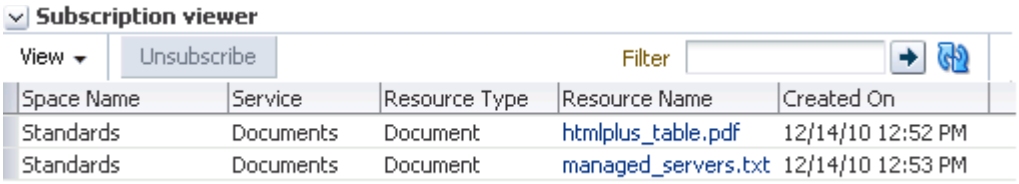
Figure 49–1 Subscription Preferences Task Flow



Users establish their object-level subscriptions by subscribing to the object directly.

Users manage their application-level subscriptions through the Subscription Preferences task flow and their object-level subscriptions through the Notifications task flow *Subscription Viewer* (Figure 49–2).

Figure 49–2 Subscription Viewer Task Flow



See Also: For more information about Notifications task flows, see Section 49.2.5.1, "Notifications Task flows."

49.2 Basic Configuration for Notifications

Notifications leverages your Mail or Worklist service connection to provide messaging channels for user notifications. If you plan to use the Mail server, then a connection to a mail server for the Mail service is required, but no further configuration is necessary for Notifications. If you plan to use the same BPEL connection for Notifications that is used by the Worklist service, you must add this configuration to the `adf-config.xml` file.

This section provides an overview of those connections and configurations and the process of adding Notifications task flows to an application page. It includes the following subsections:

- Section 49.2.1, "Adding the Notifications Namespace"
- Section 49.2.2, "Creating a Notifications Connection to a Mail Server"
- Section 49.2.3, "Creating a Notifications Connection to a BPEL Server"
- Section 49.2.4, "Setting Application Defaults for Notifications"
- Section 49.2.5, "Adding Notifications Task Flows to Your Application"
- Section 49.2.6, "Setting Security for Notifications"

49.2.1 Adding the Notifications Namespace

To establish the Notifications namespace with your application, add the following code to the `adf-config` element at the top of the file (Example 49–1):

Tip: You can find the `adf-config.xml` file in the Application Resources panel, under **Descriptors**, in the **ADF META-INF** folder.

Example 49–1 Configuring the Application to Recognize Notifications Configuration

```
xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config"
```

For example (bold) (Figure 49–2):

Example 49–2 Notifications Configuration in the adf-config Element

```
<?xml version="1.0" encoding="US-ASCII" ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
     xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
     xmlns:sec="http://xmlns.oracle.com/adf/security/config"
     xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
     xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config"
     xmlns:worklistC="http://xmlns.oracle.com/webcenter/worklist/config"
```

```
xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config"
xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
xmlns:searchC="http://xmlns.oracle.com/webcenter/search/config"
xmlns:relC="http://xmlns.oracle.com/webcenter/relationship/config"
…
```

49.2.2 Creating a Notifications Connection to a Mail Server

To enable users to select a mail messaging channel, you must configure a connection to a Mail server (for more information, see Section 36.2.2, "Setting up Connections for the Mail Service"). Once this is established, you add the following entry in the `adf-config.xml` file to associate the Mail server with Notifications (Example 49–3).

Example 49–3 Associating a Mail Server with Notifications in adf-config.xml

```
<notificationC:adf-notification-config
          xmlns="http://xmlns.oracle.com/webcenter/notification/config">
     <ConnectionType>MAIL</ConnectionType>
     <ConnectionName>someMailConnection</ConnectionName>
</notificationC:adf-notification-config>
```

Where `ConnectionName` is the name of a configured Mail server.

Example 49–4 presents a code sample, with the previous code sample in place (bold).

Example 49–4 A Mail Connection for Notifications in adf-config.xml

```
<?xml version="1.0" encoding="US-ASCII" ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
     xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
     xmlns:sec="http://xmlns.oracle.com/adf/security/config"
     xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
     xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config"
     xmlns:worklistC="http://xmlns.oracle.com/webcenter/worklist/config"
     xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config"
     xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
     xmlns:searchC="http://xmlns.oracle.com/webcenter/search/config"
     xmlns:relC="http://xmlns.oracle.com/webcenter/relationship/config"
     xmlns:collabC="http://xmlns.oracle.com/webcenter/collab/config"
     xmlns:listC="http://xmlns.oracle.com/webcenter/list/config"
     xmlns:rcs="http://xmlns.oracle.com/adf/rcs/adf-config"
     xmlns:rcv="http://xmlns.oracle.com/adf/rcs/viewer/adf-config"xmlns:tagC=
          "http://xmlns.oracle.com/webcenter/tagging/config"
     xmlns:doclibC="http://xmlns.oracle.com/webcenter/doclib/config">
<adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
     <adf-property name="adfAppUID" value="ps3stage2app-7762"/>
</adf:adf-properties-child>
<sec:adf-security-child xmlns="http://xmlns.oracle.com/adf/security/config">
     <CredentialStoreContext
          credentialStoreClass="oracle.adf.share.security.providers.jps.
          CSFCredentialStore"
          credentialStoreLocation="../../src/META-INF/jps-config.xml"/>
     <sec:JaasSecurityContext
          initialContextFactoryClass="oracle.adf.share.security.
          JAASInitialContextFactory"
          jaasProviderClass="oracle.adf.share.security.providers.jps.
          JpsSecurityContext" authorizationEnforce="true"
          authenticationRequire="true"/>
</sec:adf-security-child>
```

```
<wpsC:adf-service-config>
     <wpsC:data-source jndi-name="java:/comp/env/jdbc/WebCenterDSDS"/>
</wpsC:adf-service-config>
<notificationC:adf-notification-config
          xmlns="http://xmlns.oracle.com/webcenter/notification/config">
     <ConnectionType>MAIL</ConnectionType>
     <ConnectionName>someMailConnection</ConnectionName>
</notificationC:adf-notification-config>
<searchC:adf-search-config
          xmlns="http://xmlns.oracle.com/webcenter/search/config">
…
```

49.2.3 Creating a Notifications Connection to a BPEL Server

To furnish users with the option to select mail, phone text, or Worklist messaging channels, you must configure a connection to a BPEL server (for more information, see Section 51.3.1, "Setting up Connections for the Worklist Service"). Notifications leverages the BPEL server that is specified for the Worklist service to provide multiple messaging channels for the notifications it generates. Once a connection to a BPEL server is established, you add the following entry in the `adf-config.xml` file to associate the BPEL server with Notifications (Example 49–5).

Example 49–5 Configuring Notifications to Use the BPEL Server for Messaging

```
<notificationC:adf-notification-config
          xmlns="http://xmlns.oracle.com/webcenter/notification/config">
     <ConnectionType>BPEL</ConnectionType>
     <ConnectionName>worklistConnection</ConnectionName>
     <SenderEmailAddress>example@company.com<SenderEmailAddress>
     <SenderSMSAddress>12345</SenderSMSAddress>
</notificationC:adf-notification-config>
```

Where:

- **ConnectionName** is the name you provided for the BPEL server when you set up that connection for the Worklist service.
- **SenderEMailAddress** is a mail address from which all Notifications messages are sent. The sender mail address must match at least one driver that is configured to send messages from a corresponding domain.
- **SenderSMSAddress** is the four- to six-digit number that is used by the User Messaging Server (UMS) as the driver from which all Notifications messages are sent. The sender SMS address must match at least one driver that is configured to send messages from a corresponding domain.

Example 49–6 presents a code sample, with the previous code sample in place (bold).

Example 49–6 Notifications Entries in the adf-config.xml File

```
<?xml version="1.0" encoding="US-ASCII" ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
     xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
     xmlns:sec="http://xmlns.oracle.com/adf/security/config"
     xmlns:wpsC="http://xmlns.oracle.com/webcenter/framework/service"
     xmlns:jndiC="http://xmlns.oracle.com/adf/jndi/config"
     xmlns:worklistC="http://xmlns.oracle.com/webcenter/worklist/config"
     xmlns:notificationC="http://xmlns.oracle.com/webcenter/notification/config"
     xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config"
     xmlns:searchC="http://xmlns.oracle.com/webcenter/search/config"
```

```
     xmlns:relC="http://xmlns.oracle.com/webcenter/relationship/config"
     xmlns:collabC="http://xmlns.oracle.com/webcenter/collab/config"
     xmlns:listC="http://xmlns.oracle.com/webcenter/list/config"
     xmlns:rcs="http://xmlns.oracle.com/adf/rcs/adf-config"
     xmlns:rcv="http://xmlns.oracle.com/adf/rcs/viewer/adf-config"xmlns:tagC=
          "http://xmlns.oracle.com/webcenter/tagging/config"
     xmlns:doclibC="http://xmlns.oracle.com/webcenter/doclib/config">
<adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
     <adf-property name="adfAppUID" value="ps3stage2app-7762"/>
</adf:adf-properties-child>
<sec:adf-security-child xmlns="http://xmlns.oracle.com/adf/security/config">
     <CredentialStoreContext
          credentialStoreClass="oracle.adf.share.security.providers.jps.
          CSFCredentialStore"
          credentialStoreLocation="../../src/META-INF/jps-config.xml"/>
     <sec:JaasSecurityContext
          initialContextFactoryClass="oracle.adf.share.security.
          JAASInitialContextFactory"
          jaasProviderClass="oracle.adf.share.security.providers.jps.
          JpsSecurityContext" authorizationEnforce="true"
          authenticationRequire="true"/>
</sec:adf-security-child>
<wpsC:adf-service-config>
     <wpsC:data-source jndi-name="java:/comp/env/jdbc/WebCenterDSDS"/>
</wpsC:adf-service-config>
<notificationC:adf-notification-config
          xmlns="http://xmlns.oracle.com/webcenter/notification/config">
     <ConnectionType>BPEL</ConnectionType>
     <ConnectionName>worklistConnection</ConnectionName>
     <SenderEmailAddress>example@company.com<SenderEmailAddress>
     <SenderSMSAddress>12345</SenderSMSAddress>
</notificationC:adf-notification-config>
<searchC:adf-search-config
          xmlns="http://xmlns.oracle.com/webcenter/search/config">
…
```

49.2.4 Setting Application Defaults for Notifications

This section provides information about the XML file you use to set application-level defaults for all users for Notifications. It includes the following subsections:

- Section 49.2.4.1, "What You Should Know About Subscription Defaults"
- Section 49.2.4.2, "Setting Notifications Subscription Defaults"

49.2.4.1 What You Should Know About Subscription Defaults

Administrator-level subscription preferences are set in a custom XML file that you create and then use to supersede the file that is provided in the `notification-repository.jar` for this purpose out of the box (`notification-service-settings.xml`). The settings in the custom XML file are analogous to the application-level subscriptions settings available to users through the Subscription Preferences task flow.

Each setting provides the following attributes:

- `id`—for specifying the service ID:
  - `oracle.webcenter.peopleconnections.connections`, the Connections feature of the People Connections service

  - `oracle.webcenter.peopleconnections.wall`, the Message Board feature of the People Connections service
  - `oracle.webcenter.peopleconnections.kudos`, the Feedback feature of the People Connections service

- `subscription-enabled`—For specifying the default value for the preference option: `true` or `false`

  Tip: Rather than enabling or disabling the entire subscription capability, the `subscription-enabled` attribute merely sets the initial state of the preference option. For example, if `subscription-enabled="true"`, then the associated subscription option is checked by default in the Subscription Preferences task flow. If `subscription-enabled="false"`, then the associated subscription option is not checked by default in the task flow.

- `end-user-configurable`—For enabling users to change the established default or preventing users from doing so: `true` or `false`

These attributes work together to determine the initial state of the Subscription Preferences task flow.

Table 49–2 illustrates the effect of custom administrator-level subscriptions settings on the appearance of the Subscription Preferences task flow.

Table 49–2 Effect of Administrator Defaults on Subscriptions Preferences

subscription-enabled[1]	end-user-configurable	Option in Preferences
True	True	Rendered normally, checkbox is checked
True	False	Grayed out, checkbox is checked
False	True	Rendered normally, checkbox is clear
False	False	Hidden, checkbox is hidden

[1] Rather than enabling or disabling the entire subscription capability, the `subscription-enabled` attribute merely sets the initial state of the subscription.

Tip: The most typical scenario for most notifications is `False`, `True`, `Rendered normally, checkbox is clear`.

Table 49–3 lists the types of actions that can trigger an application-level notification and associates them with their related service ID.

Table 49–3 Application-Level Activities that Can Trigger Notifications

Activity	Related Service ID
A user sends you an invitation to connect	`oracle.webcenter.peopleconnections.connections`
A user posts a message to your Message Board	`oracle.webcenter.peopleconnections.wall`
A user likes your post on another user's Message Board	`oracle.webcenter.peopleconnections.wall`
A user comments on your post on another user's Message Board	`oracle.webcenter.peopleconnections.wall`
A user posts feedback for you	`oracle.webcenter.peopleconnections.kudos`

49.2.4.2 Setting Notifications Subscription Defaults

To set defaults for application-level Subscription preferences:

1. Navigate to a directory with a path that contains `/oracle/webcenter/notification`, and create the folder `custom`.

   Tip: The directory structure can start or end with any directory or directories, as long as it has `/oracle/webcenter/notification/custom` in the path.

2. In the `custom` folder, or in any subdirectory under `/oracle/webcenter/notification/custom/`, create the file `notification-service-settings.xml`.

3. In the XML file, enter values for all application-level subscription options.

   Example 49–7 provides sample content for an application-wide subscription preferences setting file and an example of each required option.

Example 49–7 Sample Subscriptions Settings XML File

```
<notification-service_settings xmlns="http://xmlns.oracle.com/webcenter/notification">
  <subscription-settings>
    <service id="oracle.webcenter.peopleconnections.connections" subscription-enabled="true"
      end-user-configurable="false"/>
    <service id="oracle.webcenter.peopleconnections.wall" subscription-enabled="false"
      end-user-configurable="true"/>
    <service id="oracle.webcenter.peopleconnections.kudos" subscription-enabled="false"
      end-user-configurable="true"/>
  </subscription-settings>
</notification-service_settings>
```

Note: If an option is not provided, the default values `false/false` are assigned for the service.

4. Run the WLST command `importMetadata`, and import the directory content into your metadata store.

   See Also: For information about the `importMetadata` command (and other WLST commands), see the section "importMetadata" in *Oracle Fusion Middleware WebLogic Scripting Tool Command Reference*.

   For example:

   ```
   wls: /wc_domain/serverConfig> importMetadata(application='webcenter',
     server='serverName', fromLocation='directoryPath', docs='/**')
   ```

   Where:

   - *`application`* is the name that identifies your application
   - *`serverName`* is the name of the server on which the application is running
   - *`directoryPath`* is the directory path under which `oracle/webcenter/notification/custom/<any_sub_dir_after_this>/notification-service-settings.xml` is located.

For example, if the directory path to `notification-service-settings.xml` is `/scratch/mydir/oracle/webcenter/notification/custom`, enter `/scratch/mydir` for `directoryPath`.

- *`docs`* identifies the content to be imported, in this example, the path and files that fall under those specified for `directoryPath`.

Table 49–4 describes the effect of various combinations of settings for the service ID `oracle.webcenter.peopleconnections.connections`.

Table 49–4 Effects of Subscription Configurations for Connections

subscription-enabled	end-user-configurable	Effect
true	true	■ The subscribing user receives a notification message when another user sends him an invitation to connect. ■ The user can change this default.
true	false	■ The subscribing user receives a notification message when another user sends him an invitation to connect. ■ The user cannot change this default.[1]
false	true	■ The subscribing user does not receive a notification message when another user sends him an invitation to connect. ■ The user can change this default.
false	false	■ The subscribing user does not receive a notification message when another user sends him an invitation to connect. ■ The option for changing this default is hidden.

[1] This is the out-of-the-box default

Table 49–5 describes the effect of various combinations of settings for the service ID `oracle.webcenter.peopleconnections.wall`.

Table 49–5 Effects of Subscription Configurations for Message Board

subscription-enabled	end-user-configurable	Effect
true	true	■ The subscribing user receives a notification message when another user posts a message on his Message board, likes his Message Board post, or comments on his Message Board post. ■ The user can change this default.

Table 49–5 (Cont.) Effects of Subscription Configurations for Message Board

subscription-enabled	end-user-configurable	Effect
true	false	▪ The subscribing user receives a notification message when another user posts a message on his Message board, likes his Message Board post, or comments on his Message Board post. ▪ The user cannot change this default.
false	true	▪ The subscribing user does not receive a notification message when another user posts a message on his Message board, likes his Message Board post, or comments on his Message Board post. ▪ The user can change this default.
false	false	▪ The subscribing user does not receive a notification message when another user posts a message on his Message board, likes his Message Board post, or comments on his Message Board post. ▪ The option for changing this default is hidden.

Table 49–6 describes the effect of various combinations of settings for the service ID `oracle.webcenter.peopleconnections.kudos`.

Table 49–6 Effect of Subscription Configurations for Feedback

subscription-enabled	end-user-configurable	Effect
true	true	▪ The subscribing user receives a notification message when another user leaves feedback for him. ▪ The user can change this default.
true	false	▪ The subscribing user receives a notification message when another user leaves feedback for him. ▪ The user cannot change this default.
false	true	▪ The subscribing user does not receive a notification message when another user leaves feedback for him. ▪ The user can change this default.
false	false	▪ The subscribing user does not receive a notification message when another user leaves feedback for him. ▪ The option for changing this default is hidden.

49.2.5 Adding Notifications Task Flows to Your Application

This section provides a brief overview of Notifications task flows and describes how to add them to your application pages. It includes the following subsections:

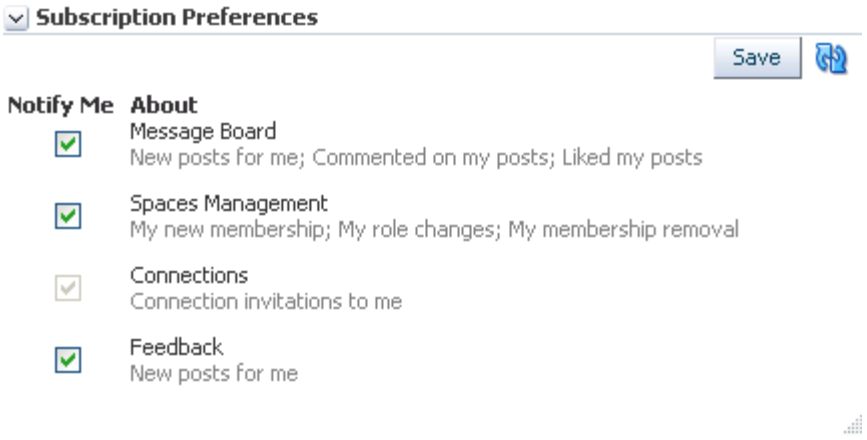
- Section 49.2.5.1, "Notifications Task flows"
- Section 49.2.5.2, "How to Add a Notifications Task Flow to a Page"

49.2.5.1 Notifications Task flows

Notification Services exposes two task flows for managing user subscriptions:

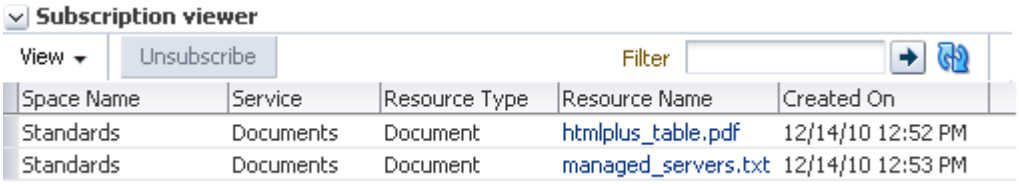
- Subscription Preference provides controls for taking, viewing, and unsubscribing from application-level subscriptions (Figure 49–3)

Figure 49–3 Subscription Preferences Task Flow



- Subscription Viewer provides controls for viewing and unsubscribing from object-level subscriptions (Figure 49–4).

Figure 49–4 Subscription Viewer Task Flow



49.2.5.2 How to Add a Notifications Task Flow to a Page

To add a Notifications task flow to a page:

1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security and create a customizable page in your application.
2. Open the page on which to add the task flow.
3. In the Resource Palette, in the My Catalogs pane, open the **WebCenter Portal - Services Catalog**, then open the **Task Flows** folder.
4. Drag the task flow onto a page region, then choose **Region** from the pop-up list.
5. In the Confirm Add ADF Library dialog, click **Add Library** to add all required libraries for the component.
6. For Subscription Preferences only, optionally, in the Edit Task Flow Binding dialog, provide a path to your input parameter map and values for the task flow's input parameters, and click **OK**.

   See Also: For information about Subscription Preferences task flow input parameters, see Section 49.2.5.3, "Subscription Preferences Task Flow Input Parameters."

   The task flow displays on the page. For example, in the Source view, you will see the content depicted in Example 49–8.

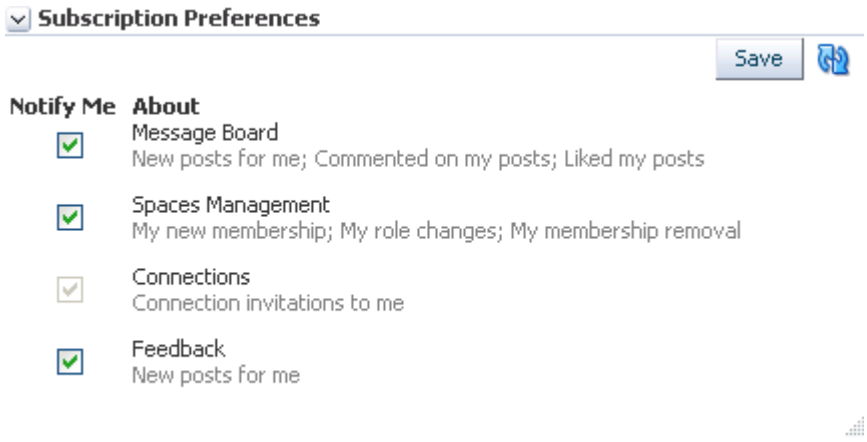
Example 49–8 Source View Content of the Subscription Preferences Task Flow

```
<af:region value="#{bindings.SubscriptionPreferences1.regionModel}" id="r1"/>
```

7. In the Confirm Add Form Element dialog, click **No**, to avoid wrapping a form element around the task flow.
8. Save your project, then run your page to a browser.

   The task flow appears in your browser (Figure 49–5).

Figure 49–5 Subscription Preferences Task Flow



49.2.5.3 Subscription Preferences Task Flow Input Parameters

The Subscription Preferences task flow has input parameters that you can use to control the behavior of a given task flow instance. Table 49–7 lists and describes the input parameters associated with this task flow.

Table 49–7 Subscription Preferences Task Flow Input Parameters

Input Parameter	Description
`hideActions`	A control for showing or hiding the **Save** button and **Refresh** icon on the task flow ▪ `${true}` to hide these controls ▪ `${false}` to show them (default)
`scopeName`	The name of the scope for which to set subscription preferences.

49.2.6 Setting Security for Notifications

Notifications must know the identity of a user to preserve the user's Notifications settings and to track the user's activities on subscribed services and objects. To that end, you must at least configure your application to authenticate users such that they have distinct identities for the purposes of personalization and user preferences.

For details on how to implement a basic security solution for your Framework application, see Section 69.3, "Configuring ADF Security." For details on how to implement a complete security solution for your Framework application, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."

After you configure ADF security for your application, you can open the `jazn-data.xml` file and modify your sample user's privileges for each task flow. To

open the ADF Security Policies Editor, locate the file in the Application Resources panel and double-click its name. You can further configure grants in this view.

Note: To successfully grant user permissions for Notifications, you must configure user permissions after deployment.

49.3 Advanced Information for Notifications

This section provides information about the APIs that are available for use with Notifications. It includes the following subsections:

- Section 49.3.1, "Using Notifications Java APIs"
- Section 49.3.2, "Using Notifications Data Controls"

49.3.1 Using Notifications Java APIs

This section provides an overview of Notifications Java APIs. It includes the following subsections:

- Section 49.3.1.1, "Configuration Settings Required to Use Notifications Java APIs"
- Section 49.3.1.2, "Introduction to Notifications Java APIs"
- Section 49.3.1.3, "How to Set up Your Application to Use Notifications Java APIs"

See Also: For detailed information about the Notifications Java APIs, see the *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

49.3.1.1 Configuration Settings Required to Use Notifications Java APIs

To use Notifications Java APIs, you must have the following entries in your `adf-config.xml` file:

```
<namespace path="/oracle/webcenter/page/scopedMD"
     metadata-store-usage="WebCenterFileMetadataStore"/>
<namespace path="/pageDefs" metadata-store-usage="WebCenterFileMetadataStore"/>
```

If these entries are not present in `adf-config.xml`, you can add them using your preferred text editor.

49.3.1.2 Introduction to Notifications Java APIs

Notifications Java APIs are located in the class `oracle.webcenter.notification`. Within the class, `SubscriptionManager` provides the interface for creating and managing user subscriptions using the following Java APIs:

- `FilterOption`—Provides the filter criteria. Use an instance of this class while querying a user's subscriptions.
- `NotificationServiceFactory`—Factory for creating and retrieving instances that correspond to Notifications interfaces. This is the entry point to the Notifications API.
- `SortOption`—Defines the sort attributes for fetching subscriptions. Use an instance of this class in `SubscriptionManager` methods to query a user's subscriptions.
- `Subscription`—Contains information about a user's subscription. Subscriptions for a user are created via the method that is exposed by `SubscriptionManager`.

Instances of this class are returned by the methods exposed in `SubscriptionManager`.

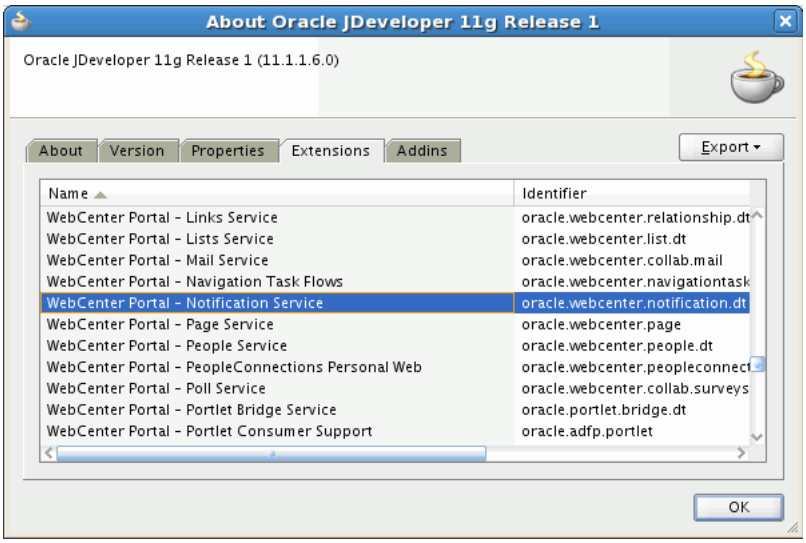
- `SubscriptionPreference`—Represents application- or scope-level subscriptions as a preference and enables setting and unsetting of subscriptions for the corresponding service.

> **See Also:** For detailed information about the Notifications Java APIs, see the *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

49.3.1.3 How to Set up Your Application to Use Notifications Java APIs

To use the Notifications Java APIs, WebCenter Portal extension must be present in your JDeveloper application. In JDeveloper, go to the **Help - About** menu, and click the **Extensions** tab. Look for the Service, as shown in Figure 49–6. For information about adding the WebCenter Portal extension, see Chapter 3, "Preparing Your Development Environment."

Figure 49–6 JDeveloper Extensions

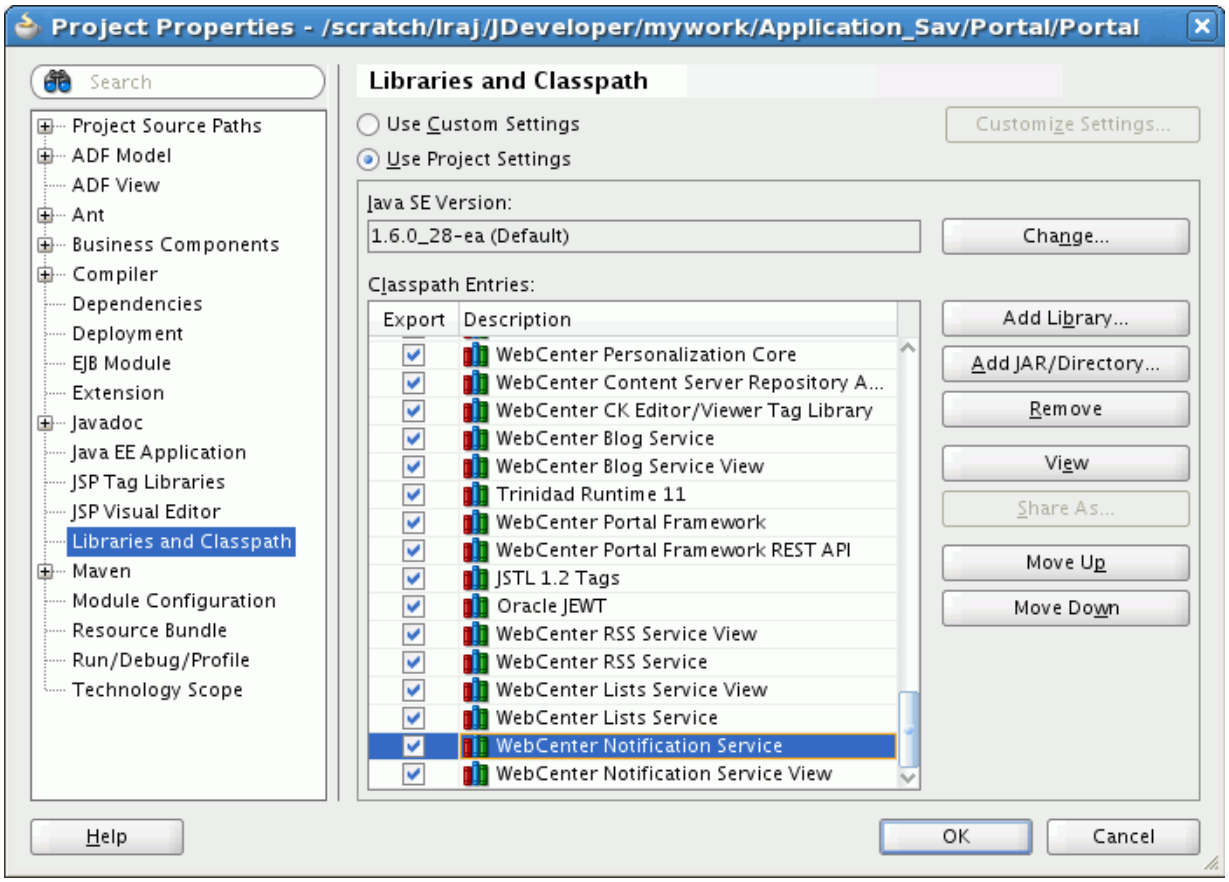


After confirming that Notifications is included with the WebCenter Portal extension, you must add the Notifications libraries to the project if they are not already present.

To add Notifications libraries to your project:

1. Right-click your project and select **Project Properties** and then **Libraries and Classpath**.
2. Click **Add Library** and select the **WebCenter Notification Service** and **WebCenter Notification Service View** libraries (Figure 49–7).
3. Click **OK**.

Figure 49–7 Libraries and Classpath in Project Properties



4. Click **OK**.

49.3.2 Using Notifications Data Controls

Notifications provides a data control that enables you to create your own visualization of the subscription management functionality. This section provides an overview of the data control **Notification Subscription Data Control**, and lists and describes its supported methods, attributes, and classes.

> **See Also:** For information about using data controls in a Framework application, see Section 7.1.3, "Using WebCenter Portal Data Controls."

This section includes the following subsections:

- Section 49.3.2.1, "Adding a Data Control to Your Project"
- Section 49.3.2.2, "Understanding the Notification Subscription Data Control"
- Section 49.3.2.3, "Notification Subscription Data Control Methods and Attributes"
- Section 49.3.2.4, "Notification Subscription Data Control Classes"

49.3.2.1 Adding a Data Control to Your Project

Add a data control to your project by right-clicking it in the Resource Palette and selecting **Add to Project** from the resulting context menu. Once added, you can browse the data control's methods and attributes by expanding it in the Data Controls panel in the Application Navigator.

Add a data control to an application page by dragging it onto the page from the Data Controls panel.

> **See Also:** For information about using data controls in a Framework application, see Section 7.1.3, "Using WebCenter Portal Data Controls."

49.3.2.2 Understanding the Notification Subscription Data Control

The Notification Subscription Data Control enables you to create a custom UI that is equivalent to the Subscription Preferences and Subscription Viewer task flows should these prove insufficient for your requirements.

You can use the data control as a means of adding application-level subscription preferences. For example, if you have developed your own preferences mechanism in your application, you can use the data control to add subscription preferences to that existing mechanism rather than using the Subscription Preferences task flow directly. Or you can build a custom subscription viewer UI, where all of the current user's object-level subscriptions are listed.

49.3.2.3 Notification Subscription Data Control Methods and Attributes

This section lists and describes the methods exposed in the Notification Subscription Data Control. It includes the following subsections:

- Section 49.3.2.3.1, "Method: getUserSubscriptionPreferences"
- Section 49.3.2.3.2, "Method: getScopeSubscriptionPreferences"
- Section 49.3.2.3.3, "Method: saveUserSubscriptionPreferences"
- Section 49.3.2.3.4, "Method: saveScopeSubscriptionPreferences"
- Section 49.3.2.3.5, "Method: getSubscriptions"
- Section 49.3.2.3.6, "Method: unsubscribe"

49.3.2.3.1 Method: getUserSubscriptionPreferences This method returns application-level subscription preferences for the current user.

Input Parameters

None

Return Values

Returns a collection of `SubscriptionPreference` objects. The `SubscriptionPreference` object indicates the user's preference for each service.

49.3.2.3.2 Method: getScopeSubscriptionPreferences This method returns the scope level subscription preferences for the current user.

This method is primarily useful in a Spaces application. In a Framework application, it is expected that the scope will always be the default scope, that is, the application. For the default scope, this method returns application-level preferences.

Input Parameters

The input parameter `scopeGuid` takes a value of type `String`. The value specifies the GUID of the scope for which the user's scope-level preferences are returned. In a Spaces application, the scope referenced here is equivalent to a *space*.

Return Values

Returns a collection of `SubscriptionPreference` objects. Each object indicates the current user's subscription preferences for each service within the specified scope.

49.3.2.3.3 Method: saveUserSubscriptionPreferences This method enables the application to commit and save changes to the user's application-level preferences. For example, if the user subscribes to all available services, then these changes can be committed by binding this method to the appropriate component on the UI (for example, an **Apply** button).

Input Parameters

None

Return Values

None

49.3.2.3.4 Method: saveScopeSubscriptionPreferences This method enables the application to commit and save changes to the user's scope-level preferences. For example, if the user subscribes to all the available services in the given scope, then these changes can be committed and saved by binding this method to the appropriate component on the UI (for example, an **Apply** button).

Input Parameters

The input parameter `scopeGUID` takes a value of type `String`. The value specifies the GUID of the scope for which the modified preferences are to be saved.

Return Values

None

49.3.2.3.5 Method: getSubscriptions The primary method for returning a list of the current user's object-level subscriptions. You can use this method to build a custom Subscription Viewer UI for showing all of a user's object-level subscriptions.

Input Parameters

Table 49–8 lists and describes the input parameters supported by this method.

Table 49–8 Input Parameters Supported by the method getSubscriptions

Input Parameter	Description
`FilterOption`	An object that contains filter conditions for querying a user's subscriptions. `FilterOption` supports filtering based on space name (Spaces only) and object name.
`SortOption`	An object that contains the sort or ordering condition for querying subscriptions. It supports sorting by object name, scope name (Spaces only), or the creation date of subscription.
`offset`	This input parameter takes an integer as a value. The offset for querying subscriptions. This is typically used while fetching the result set with pagination.
`fetchSize`	This input parameter takes an integer as a value. The maximum number of rows to be returned in one call.

Return Values

Returns a collection of `Subscription` objects matching the filter criteria and other input parameters in the order specified by the sort option.

49.3.2.3.6 Method: unsubscribe Used to unsubscribe from, or remove, a specific subscription. This method is typically used to add the unsubscribe action to the Subscription Viewer UI, where object-level subscriptions are exposed.

Note that unsubscribing from a particular service at the application and space level, or at either level, can be achieve by setting the appropriate flag in `SubscriptionPreference` and saving those via the appropriate `save*()` method listed above. The `Unsubscribe` method is useful within Subscription Viewer functionality.

Input Parameters

Table 49–9 lists and describes the input parameters supported by this method.

Table 49–9 Input Parameters Supported by the method Unsubscribe

Input Parameter	Description
`scopeGuid`	Takes a value of type `String`. The value specifies the GUID of the scope for which the user's scope-level preferences are returned. In a Spaces application, the scope referenced here is equivalent to a *space*.
`serviceId`	The ID of the unsubscribed service Valid service IDs include: ▪ `oracle.webcenter.collab.forum` ▪ `oracle.webcenter.doclib`
`objectId`	The GUID of the unsubscribed object.

Return Values

None

49.3.2.4 Notification Subscription Data Control Classes

The Notification Subscriptions Data Control exposes two classes as return values:

- `SubscriptionPreference`
- `Subscription`

49.3.2.4.1 SubscriptionPreference The class `SubscriptionPreference` represents application- or scope-level subscriptions as a preference and enables setting and unsetting of subscriptions for the corresponding service (this is followed by save call on the data control).

Table 49–10 lists and describes the properties associated with the `SubscriptionPreference` class.

Table 49–10 Properties Associated with the SubscriptionPreference Class

Property	Description
`hintsText`	A string containing the concatenation of all the applicable activities (description) within the service that generates notifications This is useful to display on the UI to tell the user what he is subscribing to.

Table 49–10 (Cont.) Properties Associated with the SubscriptionPreference Class

Property	Description
`serviceIcon`	An icon for the service
`serviceId`	ID of the corresponding service
`serviceName`	Display name of the service
`subscriptionEnabled`	A flag that indicates whether the subscription is enabled This attribute also has a setter to support the check/uncheck or select/deselect operation on the UI.
`userOverrideAllowed`	A flag that indicates whether to allow the user to override the default setting

49.3.2.4.2 Subscription The class `Subscription` is purely read-only and represents space- (Spaces only) and object-level subscriptions.

Table 49–11 lists and describes the properties associated with the **Subscription** class.

Table 49–11 Properties Associated with the Subscription Class

Property	Description
`creationDate`	The datetime this subscription was created
`objectId`	The ID of the object, if this represents object level subscription, otherwise `null`.
`objectName`	The name of the object, if this represents object level subscription, otherwise `null`.
`objectType`	The object type (as defined by the corresponding service) when this subscription represents object level subscriptions, otherwise `null`.
`scopeGuid`	The GUID of the scope or space.
`scopeName`	The display name of the scope or space.
`serviceId`	The ID of the service for this subscription object.
`serviceName`	The name of the service.
`userId`	The ID of the user who created this subscription.

50

Integrating the Recent Activities Service

This chapter describes how to integrate the Recent Activities service into your WebCenter Portal: Framework application. This service enables you to display the most recent changes to the services in your application.

This chapter includes the following sections:

- Section 50.1, "Introduction to the Recent Activities Service"
- Section 50.2, "Basic Configuration for the Recent Activities Service"
- Section 50.3, "Advanced Information for the Recent Activities Service"

For information about using the Recent Activities service, see Chapter "Working with the Recent Activities Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

50.1 Introduction to the Recent Activities Service

Oracle WebCenter Services are preconfigured to work with the Recent Activities service. Recent Activities enable users to view the most recent changes to the services in your application. For example, once you add the Documents service, it automatically produces the information that the Recent Activities service will use to display the most recent changes to the document library. This service is useful to your users who want a quick and easy way to view any additions or changes to a particular area of the application.

The Recent Activities service can track information from the following WebCenter Services:

- Documents added by the Documents service
- Announcements created by the Announcements service
- Discussion Forums created by the Discussions service
- Pages created or modified by the Page service
- List definitions created or modified by the Lists service

For more information about these services, see Chapter 33, "Integrating the Announcements Service," Chapter 34, "Integrating the Discussions Service," Chapter 30, "Integrating the Documents Service," and Chapter 19, "Enabling Runtime Creation and Management of Pages."

This section contains the following subsections:

- Section 50.1.1, "Understanding the Recent Activities Service"
- Section 50.1.2, "What Happens at Runtime"

50.1.1 Understanding the Recent Activities Service

The Recent Activities service provides a summary view of recent changes to a variety of services. You can specify the range of time to consider recent by selecting a time range from a list at the top of the task flow. Recorded activities include additions or revisions of pages, documents, discussion forums, and the like.

By default, the Recent Activities service works with the Documents, Announcements, Discussions, and Page services. You can narrow the services that the Recent Activities service tracks. For more information about this setting, see the next section, Section 50.3, "Advanced Information for the Recent Activities Service." You can also learn more about using Oracle WebCenter Portal: Services in Section 2.4, "Introducing Oracle WebCenter Portal: Services."

The Recent Activities service compares somewhat to Activity Stream (for more information, see Section 38.1.1.1, "Activity Stream"). Both track the application activities of integrated services, though Activity Stream tracks a broader range of services. For example, Recent Activities tracks the Documents, Announcements, Discussions, and Page services. Activity Stream tracks these services as well as People Connections and Wikis and Blogs. Recent Activities tracks activities no matter who performs the action. Activity Stream tracks activities performed by a user's connections and includes information about who performed the activity. Recent Activities does not include names.

The basic difference between these two services can be summarized as follows: Recent Activities provides an overview of what is happening in an application. Activity Stream provides an overview of what is happening with a user's connections.

50.1.2 What Happens at Runtime

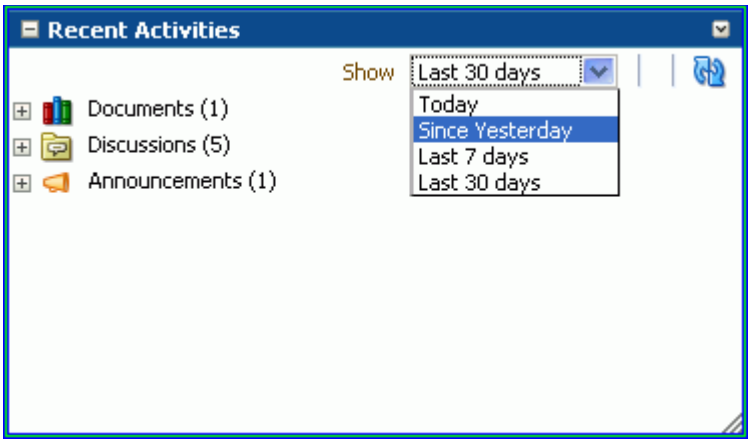
At runtime, the Recent Activities task flow fetches latest information about the services that are configured and running in your application. Figure 50–1 shows an example of the Recent Activities task flow at runtime, and displays the following activities that have occurred in the application since yesterday:

- one document has been uploaded or added using the Documents service
- five discussion threads have been modified or added using the Discussions service
- one announcement has been modified or added using the Announcements service

The time range choices (Last 30 days, Today, Since Yesterday, and so on) display based on the default values set for the Recent Activities Time Range parameter values at design time (see Table 50–1).

The Recent Activities service uses the search capabilities of WebCenter to query activities for specified periods. Search capabilities are implemented across the following WebCenter Portal service: Documents, Announcements, Discussions, and Pages. These services, if configured in an application, respond to queries of the Recent Activities service and return items pertinent to the search criteria. Responses to queries (search results) are always based on the search predicate, `DCMI_MODIFIED > [start time]`. Here `start time` is one of the time periods that you configured in the Edit Task Flow Binding dialog at design time. You can select a select a time period in the **Show** dropdown list at runtime.

Figure 50–1 Recent Activities Service at Runtime



50.2 Basic Configuration for the Recent Activities Service

This section describes how to add the Recent Activities service to your Framework application.

This section contains the following subsections:

- Section 50.2.1, "Setting up Connections for the Recent Activities Service"
- Section 50.2.2, "Adding the Recent Activities Service at Design Time"
- Section 50.2.3, "Modifying Recent Activities Service Task Flow Parameters"
- Section 50.2.4, "Setting Security for the Recent Activities Service"

50.2.1 Setting up Connections for the Recent Activities Service

You do not need to set up specific connections for the Recent Activities service. You only need to ensure that the service(s) you are tracking with the Recent Activities service is configured. For more information about setting up connections for other services, see the individual services chapters in this guide.

50.2.2 Adding the Recent Activities Service at Design Time

To use the Recent Activities service, you must add its task flow to your application. Ensure that you have at least one of the services it tracks in your application. There is a single Recent Activities task flow available in the WebCenter Portal - Services Catalog, which you can add to your application to display information about services in your application to your users.

The Recent Activities service automatically picks up other services in your application and looks for recent activities within these services. Before running your application containing the Recent Activities service, however, you may want to ensure your application contains at least one of the services you wish to monitor.

To add the Recent Activities task flow to your application:

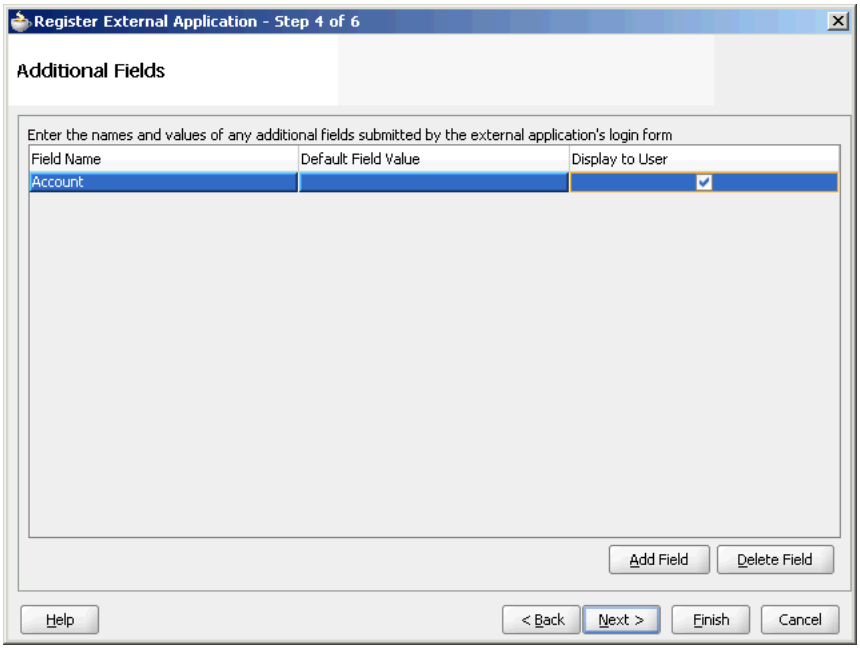
1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security and create a customizable page in your Framework application.
2. Open the source of the customizable page.

3. Ensure that your application includes at least one of the services that the Recent Activities service tracks.

   Note: If you run an application that does not contain one of these services, the Recent Activities service displays an error saying `Warning: Unable to obtain recent activity data`.

4. Create a new external application by right-clicking the **ViewController** project in your application, then choosing **New**.
5. In the New Gallery, under General, select **External Applications**, then select **External Application** from the Items list.
6. Click **OK**.
7. In the **Application Name** field, enter a name for your application, such as `MyExternalApplication`, then click **Next**.
8. On the **General** page, leave the **Login URL** field empty.
9. In the **User Name/ID Field Name** field, enter a user name. The user name should match the login credentials of the servers you are tracking. For example, if you are tracking the Documents service with Oracle Content Server, the login credentials must match those of the user on Oracle Content Server.
10. In the **Password Field Name** field, enter a password.
11. Under Authentication Details, choose **POST**, then click **Next**.
12. On the **Additional Fields** page, click the **Add Field** button.
13. For the Field Name, enter `Account`.
14. Select the **Display to User** checkbox.

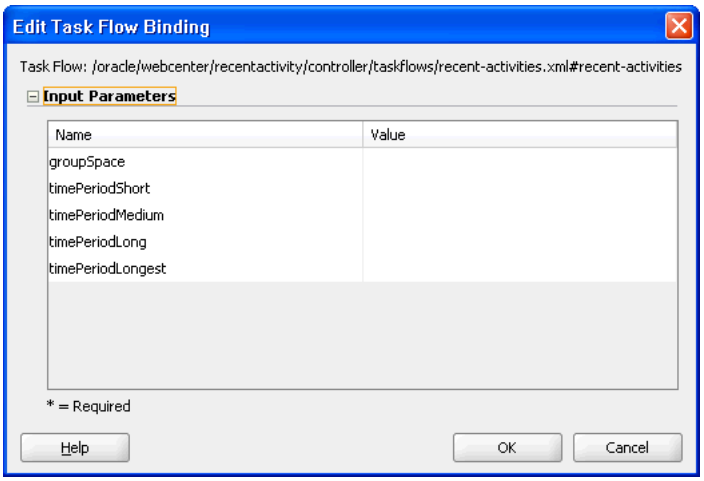
Figure 50–2 Additional Fields Page



15. Click **Finish**.

16. In the Resource Palette, open the **WebCenter Portal - Services Catalog**, then open the **Task Flows** folder.
17. Click **Recent Activities** and drag it onto your page after the `<af:form>` tag, and choose **Region**.

Figure 50–3 Edit Task Flow Binding Dialog



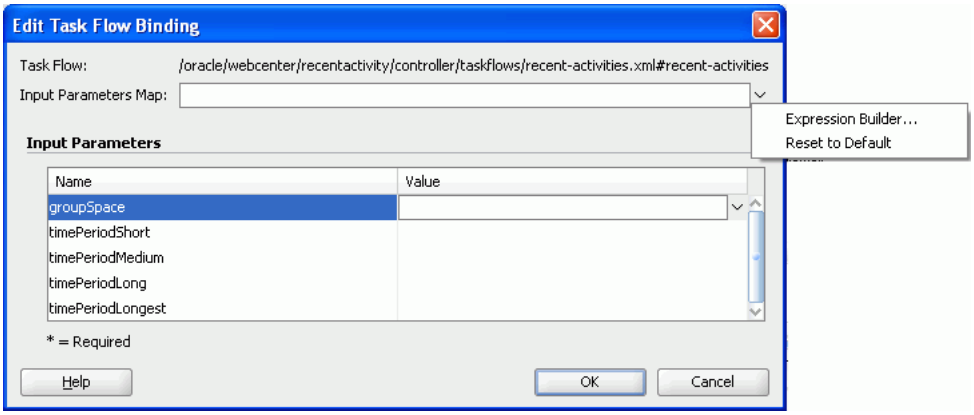
In the Edit Task Flow Binding dialog shown in Figure 50–4, you can set the values for the time range that displays at runtime. This time range controls what activities display, as you will see in the next section. If you leave these parameters empty, the Recent Activities service will use the default values of the service.

Table 50–1 describes the possible values for these parameters. The default values specified in this table come from the `adf-config.xml` file, as described in Section 50.3.1, "Refining the Behavior of the Recent Activities Service."

Table 50–1 Recent Activities Time Range Parameter Values

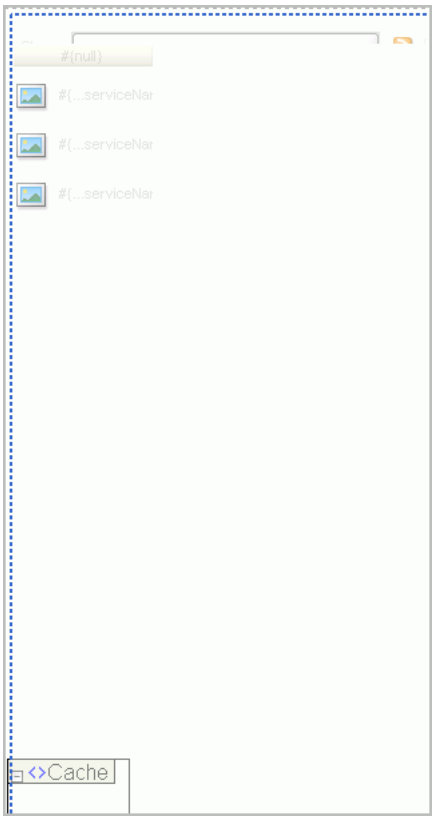
Parameter	Values
`groupSpace`	Leave the value of this parameter as `null`. Note: This parameter is only used within the Spaces application and is not relevant in non-scoped applications.
`timePeriodShort`	Default value: `Today` Possible values: `Today`, `Yesterday`, or a `Time in Minutes`
`timePeriodMedium`	Default value: `Since Yesterday` Possible values: `Today`, `Yesterday`, or a `Time in Minutes`
`timePeriodLong`	Default value: `10080` (10080 minutes is equivalent to 7 days.) Possible values: `Today`, `Yesterday`, or a `Time in Minutes`
`timePeriodLongest`	Default value: `43200` (43200 minutes is equivalent to 30 days.) Possible values: `Today`, `Yesterday`, or a `Time in Minutes`

Figure 50–4 Example of the Edit Task Flow Binding Dialog for the Recent Activities Service



18. Click **OK**. The binding displays on your page, as shown in Figure 50–5.

Figure 50–5 The Recent Activities Task Flow on a Page in the Design View



19. Save your project and run your page to the browser.

Note: When you add the Recent Activities task flow to your page, you will see a task flow parameter called **groupSpace**. The default value of this parameter is `null`, which tells the task flow to search the entire application for recent activities.

50.2.3 Modifying Recent Activities Service Task Flow Parameters

The Recent Activities service task flows have required and optional task flow binding parameters. You can adjust the parameter values when you drop the task flows onto a page or after you have placed a task flow on a page. For information about modifying parameters, see Section 33.2.2.3, "How to Modify Announcements Service Task Flow Parameters." The procedure for modifying parameters is identical across all task flows.

50.2.4 Setting Security for the Recent Activities Service

The Recent Activities service uses the security applied to the underlying services being analyzed. So, the information returned by the services only contains content for which the current user has at least View privileges. For example, a user who does not have View privileges to view content in a document library will not see recently added documents in the Recent Activities view. Similarly, a public user will only see activities returned on services exposed to the public, and do not require authentication to view.

50.3 Advanced Information for the Recent Activities Service

This section explains that in addition to the task flow parameters, you can further refine the behavior of the Recent Activities service. It also describes how you can obtain RSS news feed for your Recent Activities service.

This section contains the following subsections:

- Section 50.3.1, "Refining the Behavior of the Recent Activities Service"
- Section 50.3.2, "Obtaining RSS News Feed URL for the Recent Activities Service"

50.3.1 Refining the Behavior of the Recent Activities Service

You can change a variety of the Recent Activities service parameters after you have added it to your application. When you add the task flow to your page, you also automatically add the following entry to the `adf-config.xml` file, located in the Application Resources panel, under **Descriptors**, in the **ADF META-INF** folder:

```
  <recentactivityC:adf-recent-activity-config
 xmlns="http://xmlns.oracle.com/webcenter/recentactivity/config">
     <display-properties numServiceRows="25" shortPeriod="TODAY"
                         mediumPeriod="YESTERDAY" longPeriod="10080"
                         longestPeriod="43200"/>
    <services-filter>
      <exclude/>
    </services-filter>
  </recentactivityC:adf-recent-activity-config>
```

You can change the settings described in Table 50–2 in this entry.

Table 50–2 Recent Activities Settings in the adf-config.xml File

Setting	Description
`numServiceRows`	Default value: `25` The maximum number of rows displayed for a single service. If there are more recent activity results than this, the most recent ones up to this number are returned. For example, if there are 32 recently modified documents and this setting is 25 then only the most recent 25 will show up in the results.

Table 50–2 (Cont.) Recent Activities Settings in the adf-config.xml File

Setting	Description
`shortPeriod`	Default value: `Today` See Table 50–1. This setting is the same as the `timePeriodShort` task flow parameter, and is used when the task flow parameter is not specified.
`mediumPeriod`	Default value: `Yesterday` See Table 50–1. This setting is the same as the `timePeriodMedium` task flow parameter and is used when the task flow parameter is not specified.
`longPeriod`	Default value: `10080` (Last 7 Days) See Table 50–1. This setting is the same as the `timePeriodLong` task flow parameter and is used when the task flow parameter is not specified.
`longestPeriod`	Default value: `43200` (Last 30 Days) See Table 50–1. This setting is the same as the `timePeriodLongest` task flow parameter and is used when the task flow parameter is not specified.

`services-filter`	Default value: `null`
	Use this tag to filter out services you do not wish to track with this service. The following example omits any changes to the Page service from displaying in the Recent Activities service:
	`<services-filter>`
	`  <exclude>oracle.webcenter.page</exclude>`
	`</services-filter>`
	By default, no services are omitted.

50.3.2 Obtaining RSS News Feed URL for the Recent Activities Service

You can expose WebCenter Portal: Spaces functionality in a Framework application. Your Framework application users can find out what is happening in a specific group space through RSS news feeds.

Configure RSS news feeds for the Recent Activities service to enable users to view group space recent activities from within a portal. To obtain the group space RSS news feed URL for the Recent Activities service, use either of the following WebCenter Portal: Spaces APIs:

- `getServiceRssFeedURL`
- `GetServiceRssFeedURLbyGUID`

To obtain an RSS feed URL, you must identify the group space (by name or GUID) and specify the service required (by service ID). The service ID for the Recent Activities service is `GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID`. For information about how to use these APIs, see Section 57.2.5.3.9, "Retrieving RSS Feed URLs for Space Services."

51

Integrating the Worklist Service

This chapter describes how to integrate the Worklist service into your application. Worklists enable users to view and take action on all tasks and notifications from a BPEL (Business Process Execution Language) server all in one place.

This chapter includes the following sections:

- Section 51.1, "Introduction to the Worklist Service"
- Section 51.2, "Roadmap - Configuring the Worklist Service for WebCenter Portal: Framework Applications"
- Section 51.3, "Basic Configuration for the Worklist Service"
- Section 51.4, "Advanced Information for the Worklist Service"

For information about managing and using the Worklist service, see:

- Chapter "Managing the Worklist Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- Chapter "Working with the Worklist Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

51.1 Introduction to the Worklist Service

The Worklist service enables you to show Business Process Execution Language (BPEL) Worklist items assigned to the currently authenticated user. The BPEL Worklist items are open BPEL tasks from one or more BPEL Worklist Repositories to which your application is connected. The Worklist displays BPEL Worklist items that are a result of a task invoked as part of a BPEL Workflow process, or are a result of a message being sent to the Worklist channel on the Oracle User Messaging Service.

This section contains the following subsections:

- Section 51.1.1, "Understanding Worklists"
- Section 51.1.2, "Requirements for the Worklist Service"
- Section 51.1.3, "What Happens at Runtime"

51.1.1 Understanding Worklists

The Worklist service provides a personal, at-a-glance view of business processes that require your users' attention. These can include a request for document review or other types of business processes that come directly from your enterprise applications.

Worklist items come from a variety of sources. Some Worklist items are kicked off by events that are associated with an externally defined workflow. A workflow maps the

route an item follows once an event kicks off. This type of workflow is defined in a Worklist BPEL server, such as Oracle BPM Worklist.

Messages, alerts, and notifications might also come from the User Messaging Service (UMS). The Worklist task flow includes a control for accessing messaging preferences on this server. Use these controls to specify the channels over which to receive Oracle User Messaging Service messages and to define messaging filters.

> **See Also:** *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* for more information about the service at runtime.

51.1.2 Requirements for the Worklist Service

To use the Worklist service, you must have a BPEL server installed. For information on installing BPEL for the Worklist service, see the "Back-End Requirements for the Worklist Service" in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

For the Worklist to properly function, it must reside on a secured page, as the service uses the currently authenticated user to access the BPEL server. If the page or application that contains the Worklist service is not secured, you will encounter exception errors.

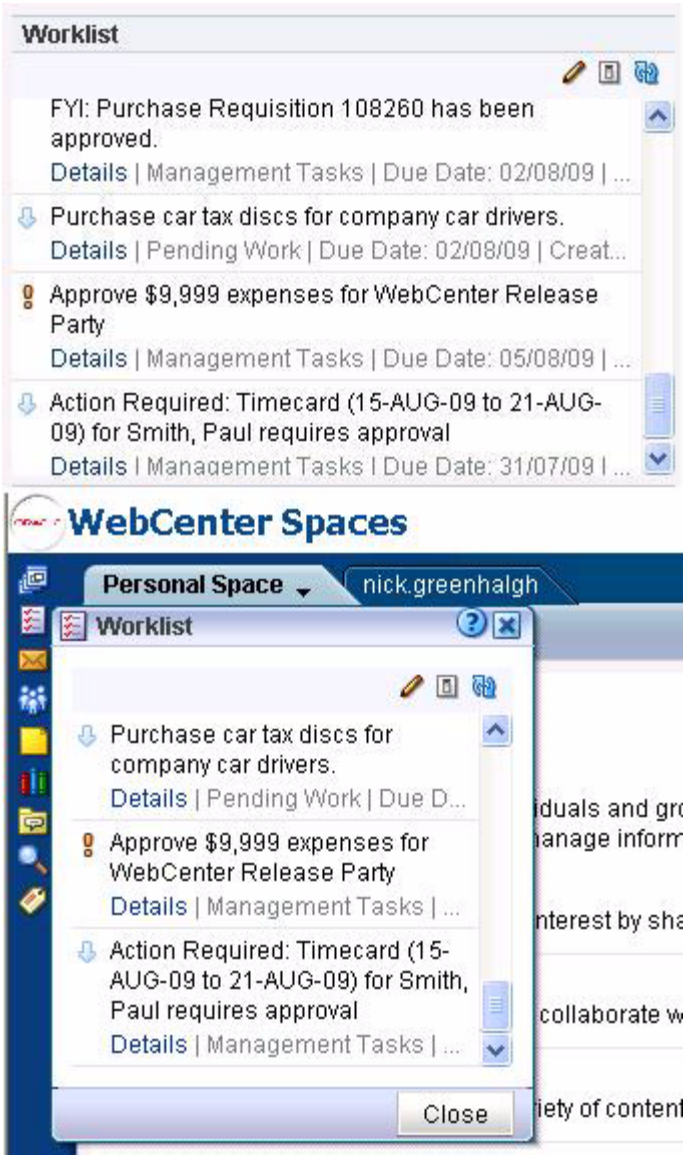
51.1.3 What Happens at Runtime

At runtime, the Worklist task flow fetches latest Worklist workflow items from BPEL workflows configured in the BPEL server. For more information, see the chapter "Working with the Worklist Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

To get a list of the items assigned to a user, the Worklist service uses the WebService access function of the public BPM client API. The Worklist service uses SAML authentication to authorize the logged-in user to fetch the list of latest items from the BPEL repository. The Worklist service fetches 25 most recent items for each Worklist connection, and determines the order of these items by using a predicate on the `TaskQueryService` API. For more information, see "Building a Custom Worklist Client" in *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

Figure 51–1 shows the Worklists service at runtime.

Figure 51–1 Worklist Service at Runtime



51.2 Roadmap - Configuring the Worklist Service for WebCenter Portal: Framework Applications

Figure 51–2 and Table 51–1 in this section provide an overview of the prerequisites and tasks required to get the Worklist service working in Framework applications.

Figure 51–2 Configuring the Worklist Service for WebCenter Portal: Framework Applications

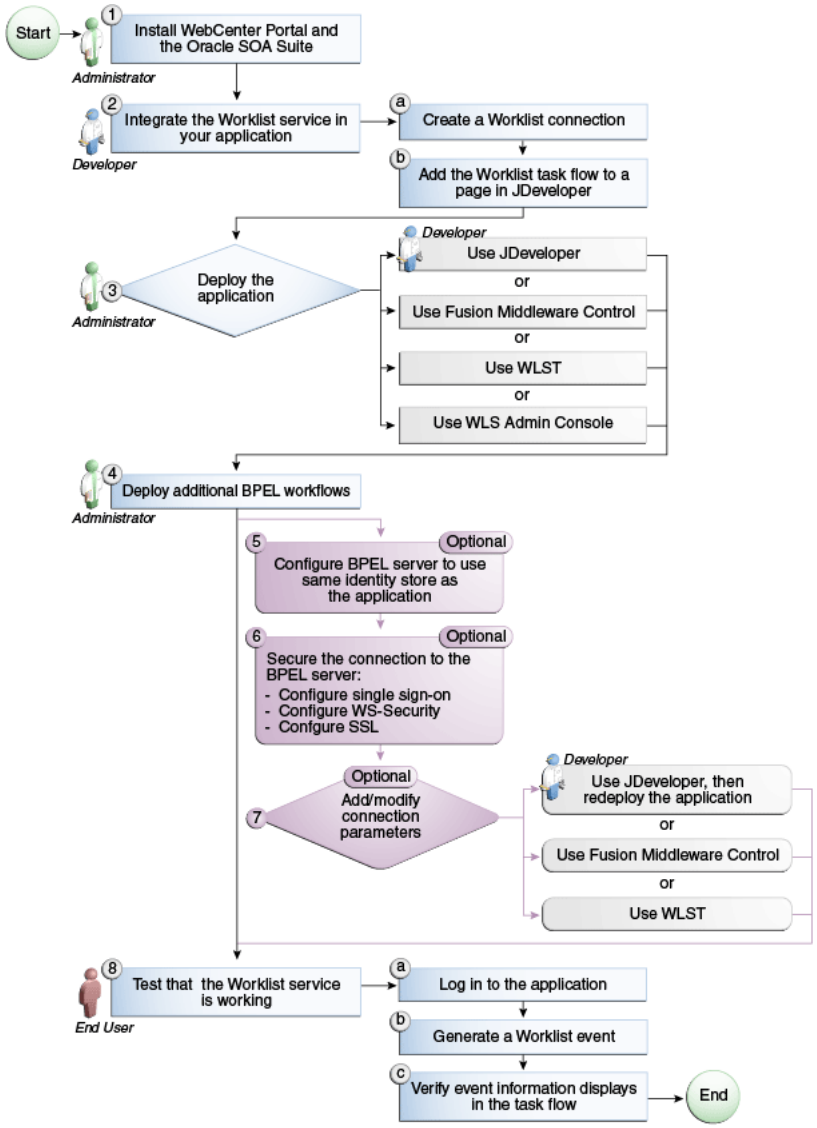


Table 51–1 Configuring the Worklist Service for WebCenter Portal: Framework Applications

Actor	Task	Sub-Task
Administrator	**1.** Install Oracle WebCenter Portal and the Oracle SOA Suite	
Developer	**2.** Integrate the Worklist service in your Framework application	**2.a** Create a Worklist connection **2.b** Add the Worklist task flow to a page in JDeveloper

Table 51–1 (Cont.) Configuring the Worklist Service for WebCenter Portal: Framework Applications

Actor	Task	Sub-Task
Developer/Administrator	**3.** Deploy the Framework application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Console (Administrator)	
Administrator	**4.** Deploy additional BPEL workflows using the Oracle BPM Worklist application	
Administrator	**5.** (Optional): Configure BPEL server to use same identity store as the application	
Administrator	**6.** (Optional): Secure the connection to the BPEL server	**6.a** Configure single sign-on **6.b** Configure WS-Security **6.c** Configure SSL
Developer/Administrator	**7.** (Optional): Add/modify connection parameters using one of the following tools: ■ JDeveloper, then redeploy the application (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator)	
End User	**8.** Test that the Worklist service is working	**8.a** Log in to the Framework application **8.b** Generate a worklist event **8.c** Verify event information displays in the task flow

51.3 Basic Configuration for the Worklist Service

This section describes how to set up a connection for the Worklist service and add the task flow to your Framework application.

This section contains the following subsections:

- Section 51.3.1, "Setting up Connections for the Worklist Service"
- Section 51.3.2, "Adding the Worklist Service at Design Time"
- Section 51.3.3, "Setting Security for the Worklist Service"

51.3.1 Setting up Connections for the Worklist Service

To use the Worklist service, you must establish a connection to one or more BPEL servers. This section describes these connections and how to create them in a Framework application.

51.3.1.1 Worklist Service Connections

The Worklist service relies on a BPEL server. Once you create a connection using the Create Worklist Connection dialog, the connection is registered in your application's `connections.xml` and referenced in your application's `adf-config.xml`. You can see the connections to your BPEL Worklist Repositories in the Application Resources pane.

> **Note:** While you can set up the connections to back-end servers at design time in Oracle JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

51.3.1.2 How to Set Up Connections for the Worklist Service

The Worklist service provides a task flow that you can add to your application at design time or at runtime. Before you can run the task flow, you create a connection from your application to a BPEL server.

> **Note:** The application you use must be prepared for service consumption. For information about doing so, see Chapter 7, "Preparing Your Application for WebCenter Portal Services."

To set up the connection for the Worklist service:

1. In the Application Resources panel for your application, right-click **Connections**, then choose **New Connection> Worklist**.
2. In the Create Worklist Connection dialog, choose whether to create the connection in the Application Resources (only the current application can use this connection) or IDE Connections (other applications you create can use the same connection).
3. In the **Connection Name** field, enter a name for your connection. This name will be used as the display name of the Worklist server.

   > **Note:** At runtime, when users view their worklist items, they can choose to group the items by "Worklist Server." The Worklist Server name used is the name you enter in the Connection Name field.

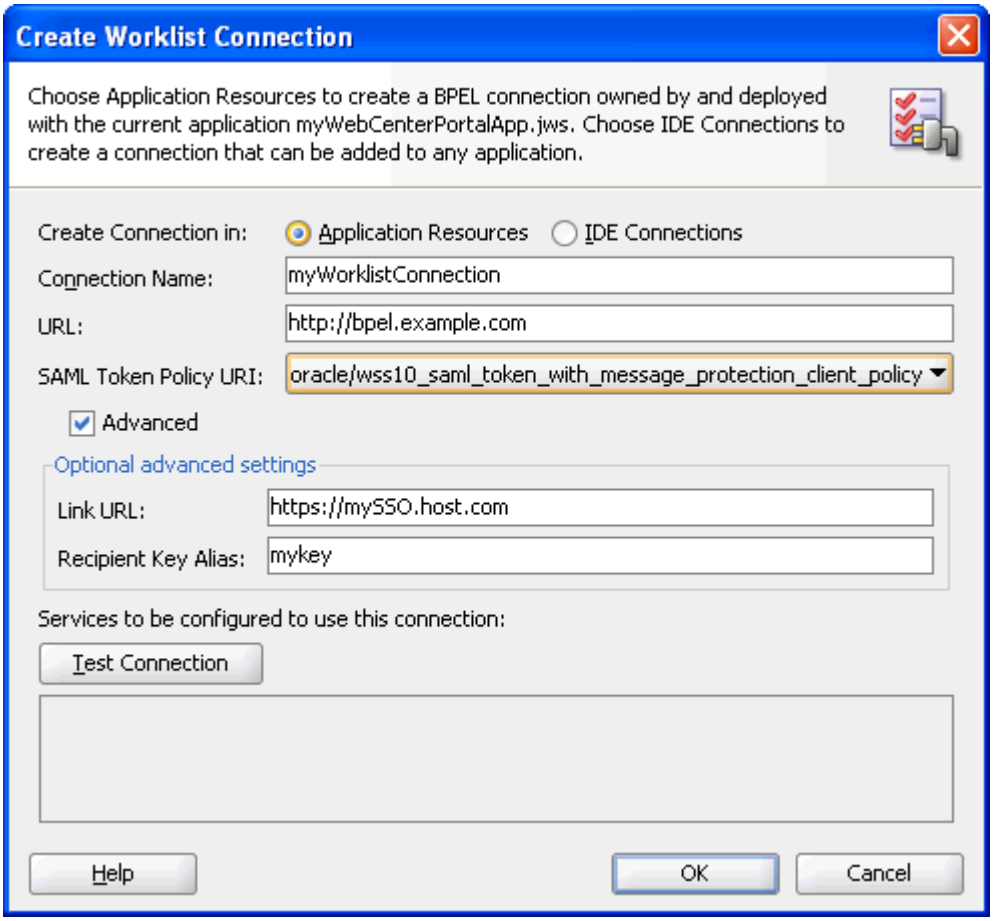
4. In the **URL** field, enter the location of your BPEL/SOA server, such as `http://bpel.example.com`. This is the URL for the managed server running the SOA processes.
5. Select the **SAML Token Policy URI** to use for this server (Figure 51–3).

   SAML (Security Assertion Markup Language) is an XML-based standard for passing security tokens defining authentication and authorization rights. An attesting entity (that already has trust relationship with the receiver) vouches for the verification of the subject by method called sender-vouches. Options available are:

   - SAML Token Client Policy (**oracle/wss10_saml_token_client_policy**) - Select to verify your basic configuration without any additional security. This is the default setting.

- SAML Token With Message Client Policy (**oracle/wss10_saml_token_with_message_protection_client_policy**) - Select to increase the security using SAML-based BPEL Web Services. If selected, you must configure keys stores both in your Framework application and in the BPEL application. For more information about this configuration, see the "Configuring Policies" in the *Oracle Application Server Web Services Security Guide*.

Figure 51–3 Create Worklist Connection Dialog



6. To enable advanced settings, select **Advanced**:

   - In the **Link URL** field, specify the URL used to link to the BPEL server. Only required if it is different to the BPEL SOAP URL, for example, when SSO or HTTPS is configured. Use the format: `protocol://host:port`.

     For example, `http://mySSO.host.com:7777`.

     This URL is used to build SSO- or HTTPS-enabled URL links to worklist items. Whereas the URL property of the Worklist service is used to access the BPEL Task Query Service, which for performance reasons does not need to go through HTTPS or an SSO server.

   - In the **Recipient Key Alias** field, enter the recipient key alias to be used for message protected SAML policy authentication. Only required when the BPEL server connection is using a SAML token policy for authentication and the application's Worklist service is using multiple BPEL server connections. For information about working with recipient key alias and WS-security, see the

chapter "Configuring WS-Security" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

7. Click **Test Connection** to validate the URL.
8. Click **OK**.

   The new connection displays in the Application Resources panel under **Connections**.

51.3.2 Adding the Worklist Service at Design Time

This section describes the Worklist task flow and how to add it to your application.

51.3.2.1 Worklist Service Task Flows

There is one task flow associated with the Worklist service, called "Worklist." You can find this task flow in the WebCenter Portal - Services Catalog in the Resource Palette.

51.3.2.2 How to Add the Worklist Service to your Application

You can add the Worklist service to your application before or after you have created your connection. However, you must set up your connection before you run the service.

To add the Worklist task flow to your application:

1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security and create a customizable page in your application.
2. Open the page where you wish to add the Worklist task flow.
3. In the Resource Palette, in the My Catalogs pane, open the **WebCenter Portal - Services Catalog**, then open the **Task Flows** folder.
4. Drag the worklist task flow onto the desired region on the page, then choose **Region** from the pop-up list. The task flow displays on the page. For example, in the Source view, you will see Figure 51–4.

Figure 51–4 Worklist Task Flow in the Page Source

```
<f:facet name="first">
    <af:region value="#{bindings.worklist1.regionModel}"
               id="worklist1"/>
  </f:facet>
```

5. Save your project, then run your page to a browser. The Worklist service displays in your browser.

51.3.3 Setting Security for the Worklist Service

The Worklist service displays the tasks for the currently authenticated user. So, for users to store and retrieve Worklist tasks on the BPEL server from your application, you must set up security on the application (as described in Section 7.2.1.1, "Implementing Security for Services"). The user names need to either exist in a shared user directory (LDAP), or set up similarly (same user name) on both the Framework application and the BPEL Server. For example, if the user `rsmith` needs to use the Worklist service in your Framework application to store and retrieve tasks from the

BPEL server, you must ensure that the user `rsmith` exists on both the BPEL server and within your application.

To enable users to perform actions on assigned Worklist tasks, you can configure SSO. If you do not configure SSO, the users will be prompted to log into the Worklist Task Details page on the BPEL server. For more information, see the *Oracle Containers for J2EE Security Guide*.

Only authenticated users can view Worklist tasks on the BPEL server to which they have permissions to view. If the application end user does not supply appropriate credentials, or if the page containing the Worklist service is not secured, she will not see any Worklist items stored on the BPEL server and exception errors may occur.

51.4 Advanced Information for the Worklist Service

The Worklist service collates all the Worklist items for the authenticated user from all connections in your application defined for the Worklist service. You can use the create connection method described in Section 51.3.1.2, "How to Set Up Connections for the Worklist Service" to create additional connections, or right-click the connection name in the Application Resources list and choose **Delete** to remove a connection.

52 Integrating the Events Service

This chapter explains how to integrate the Events service into a WebCenter Portal: Framework application at design time.

This chapter includes the following sections

- Section 52.1, "Introduction to the Events Service"
- Section 52.2, "Basic Configuration for the Events Service"
- Section 52.3, "Using the Events Service REST APIs"
- Section 52.4, "Troubleshooting the Events Service"

For information about managing and including events, see:

- "Managing the Events Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- "Working with the Events Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

52.1 Introduction to the Events Service

In Framework applications, the Events service provides access to your personal Microsoft Exchange calendar.

This section includes to following subsections:

- Section 52.1.1, "Understanding the Events Service"
- Section 52.1.2, "Requirements for the Events Service"
- Section 52.1.3, "What Happens at Runtime"

52.1.1 Understanding the Events Service

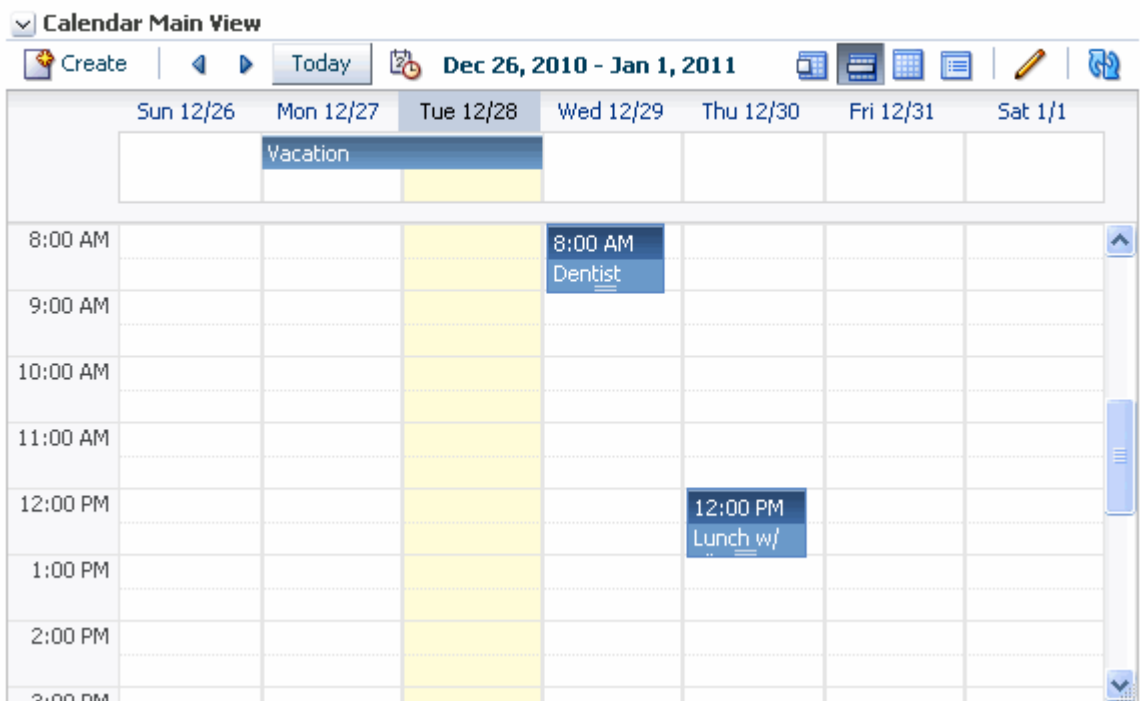
With the Events service, users can:

- Display their Microsoft Exchange calendars in the Framework application.
- Create, edit, and delete events in their Exchange calendars.
- Change the view of the Events task flow to view calendar events by month, week, or day, or as a list.
- Personalize the Events task flow to display events based on the start time, secondary time zone, list type, and list count

The Events service is integrated with many WebCenter Portal services, such as Links, Mail, and Search.

Figure 52–1 shows an example of a personal calendar in a Framework application.

Figure 52–1 Personal Calendar



52.1.2 Requirements for the Events Service

The Events service integrates with Microsoft Exchange Server to provide access to personal calendars. You must install and configure the appropriate server.

For information about installing and configuring Exchange Server, see "Events Service Prerequisites" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

52.1.3 What Happens at Runtime

At runtime, users can view events from their Microsoft Exchange calendars.

For more information about the service at runtime, see the chapter "Working with the Events Service" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Note: *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces* discusses both personal events and space events. Only the information on personal events applies to Framework applications.

52.2 Basic Configuration for the Events Service

This section includes the following subsections:

- Section 52.2.1, "Configuration Roadmap for the Events Service"
- Section 52.2.2, "Setting Up a Connection for the Events Service"

- Section 52.2.3, "Adding the Events Service at Design Time"
- Section 52.2.4, "Setting Security for the Events Service"

52.2.1 Configuration Roadmap for the Events Service

The flow chart (Figure 52–2) and table (Table 52–1) in this section provide an overview of the prerequisites and tasks required to get the Events service working in Framework applications.

Figure 52–2 Configuring the Events Service for WebCenter Portal: Framework Applications

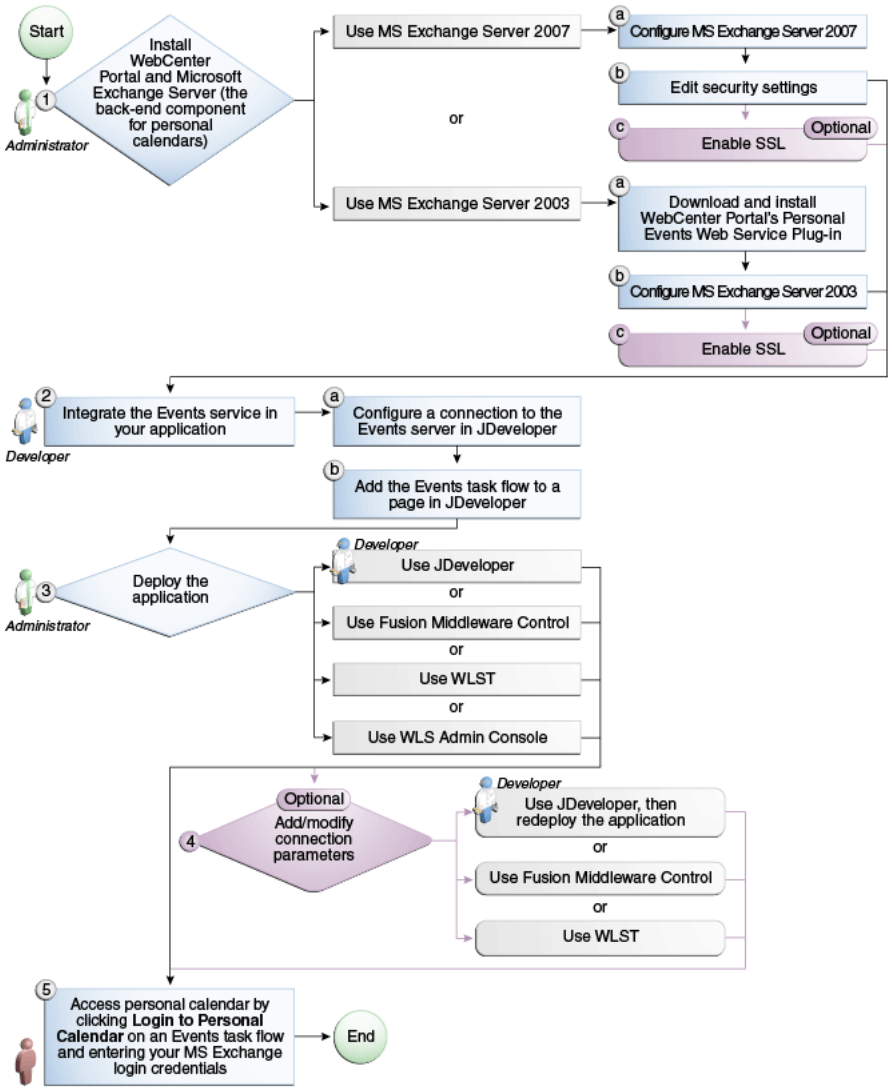


Table 52–1 Configuring the Events Service for WebCenter Portal: Framework Applications

Actor	Task	Sub-task	Notes
Administrator	**1.** Install WebCenter Portal and Microsoft Exchange Server		MS Exchange Server is the back-end component for personal calendars
	■ Use MS Exchange Server 2007	**1.a** Configure MS Exchange Server 2007 **1.b** Edit security settings **1.c** (Optional) Enable SSL	
	■ Use MS Exchange Server 2003	**1.a** Download and install WebCenter Portal's Personal Events Web Service Plug-in **1.b** Configure MS Exchange Server 2003 **1.c** (Optional) Enable SSL	
Developer	**2.** Integrate the Events service in your application	**2.a** Configure a connection to the events server in JDeveloper **2.b** Add an Events task flow to a page in JDeveloper	
Developer/ Administrator	**3.** Deploy the application using one of the following tools: ■ JDeveloper (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator) ■ WLS Admin Console (Administrator)		
Developer/ Administrator	**4.** Add/modify connection parameters using one of the following tools: ■ JDeveloper, then redeploy the application (Developer) ■ Fusion Middleware Control (Administrator) ■ WLST (Administrator)		
End User	**5.** Access personal calendar		Click **Login to Personal Calendar** on the Events task flow and enter your MS Exchange login credentials

52.2.2 Setting Up a Connection for the Events Service

To take advantage of the Events service, you must first create a connection to the Exchange server from your Framework application. To do this, ensure that you have the connection information for the server.

You can register several server connections for the Events service, but only one connection is active at a time.

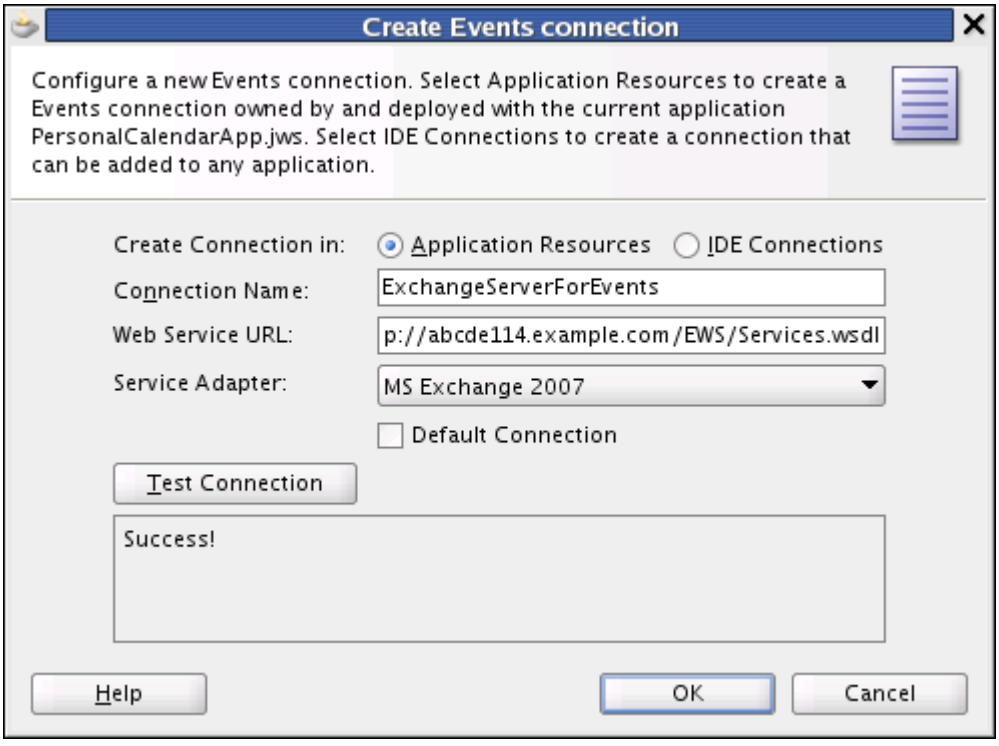
Note: While you can set up the connections to back-end servers at design time in JDeveloper, you can later add, delete, or modify connections in your deployed environment using Enterprise Manager Fusion Middleware Control. For more information, see the section "Registering Events Servers Using Fusion Middleware Control" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To set up a connection for the Events service:

1. In JDeveloper, open the application in which you plan to consume the Events service.
2. To create a connection that is available to all applications, in the Resource Palette, click the **New** icon and choose **New Connection** and then **WebCenter Personal Event Connection**.

   To create a connection that is available to the current application only, in the Application Resources panel of the Application Navigator, right-click **Connections** and choose **New Connection** and then **WebCenter Personal Event Connection**.
3. In the Create Events Connection dialog, in the **Connection Name** field, enter a meaningful name, for example `ExchangeServerForEvents` (Figure 52–3).

Figure 52–3 Create Events Connection Dialog



4. In the **Web Service URL** field, enter the URL of the web service exposing the event application. Use the format:

   protocol`://`*host*`:`*port*`/`*appWebServiceInterface*`/`*WSName*

   For example:

```
http://myexchange.com:80/ExchangeWS/PersonalEventsWebService.asmx
http://myexchange.com:80/EWS/Services.wsdl
```

5. From the **Service Adapter** dropdown list, select the version of the Microsoft Exchange server to connect to:
   - **MSExchange2003**
   - **MSExchange2007**
6. Select **Default Connection** to use this connection for the Events service. You can have multiple connections, but only one can be active (default).

   Note: After you create a connection as the active connection, you cannot edit it so that it is not the default. To use a different active connection, you must create a new connection and mark that as the default connection.

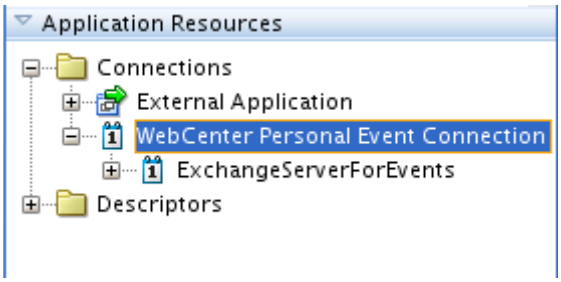
7. Click **Test Connection**, and if it is successful, then click **OK**.

   If the test is not successful, check the settings and try again.
8. If you created the connection in the Resource Palette, you must add it to the application.

   In the Resource Palette, under IDE Connections, right-click the connection and choose **Add to Application**.

   The connection is listed in the Application Resources panel of the Application Navigator (Figure 52–4). An external application connection is also created to store the user names and passwords of users who connect to the Exchange server.

Figure 52–4 Application Resources - Personal Event Connection



52.2.3 Adding the Events Service at Design Time

This section includes the following subsections:

- Section 52.2.3.1, "Events Task Flows"
- Section 52.2.3.2, "How to Add the Events Service to Your Application"

52.2.3.1 Events Task Flows

The Events service task flows enable you to add a main view or quick view of events to a page.

Table 52–2 Events Service Task Flows

Task Flow	Description
Calendar Main View	This task flow displays a calendar of events and provides users with options to view events in day, week, month, or list view.
Calendar Mini View	This task flow displays a calendar, similar to Calendar Main View, however, it requires less space. Only users with permission to edit the page and task flow can configure the event view (available in day or list view).

52.2.3.2 How to Add the Events Service to Your Application

To add an Events task flow to your application:

1. Follow the steps described in Section 7.2, "Preparing Your Framework Application to Consume Services," to implement security and create a new customizable page in your application.
2. Open the page on which you want to add the task flow.
3. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
4. Drag **Calendar Main View** or **Calendar Mini View** and drop it onto the page, inside the `af:form` tags.
5. When prompted, select **Region** as the way to create the task flow.
6. The Events task flows are contained within the `event-view.jar` library, so click **Add Library** to add the library to your project. This operation may take a moment to complete.
7. Click **OK**.
8. Save and run your page.

   Figure 52–5 shows the Calendar Main View task flow at runtime.

Figure 52–5 Calendar Main View Task Flow

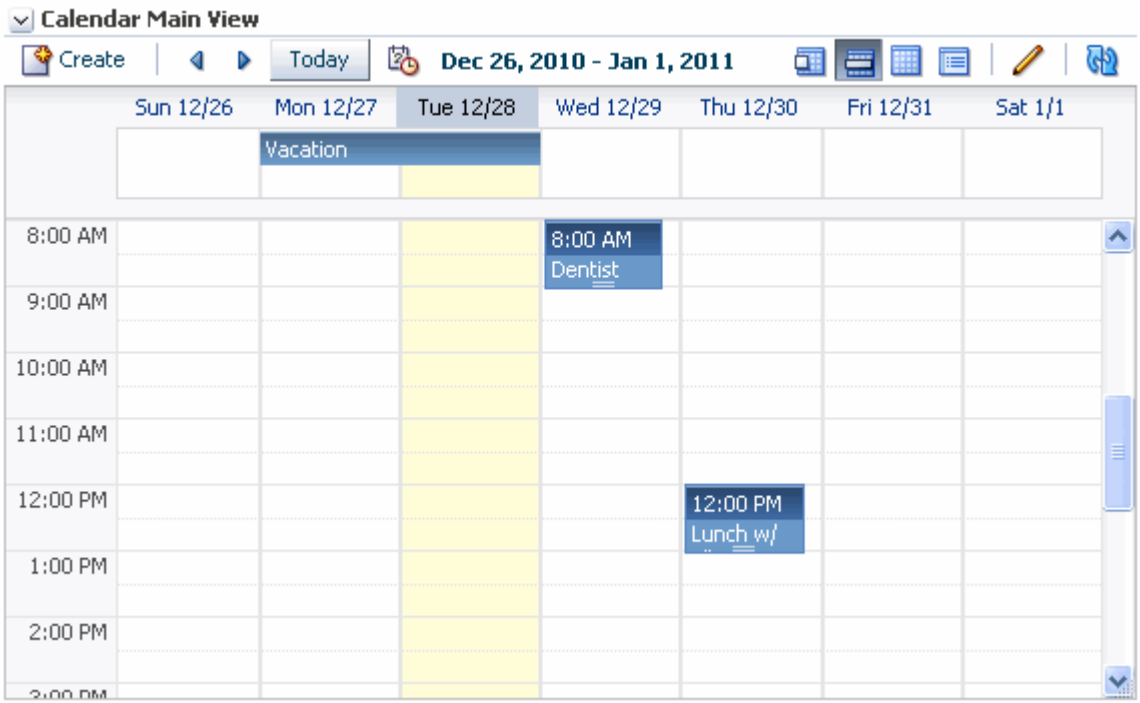
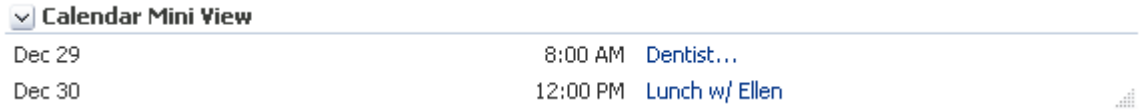


Figure 52–6 shows the Calendar Mini View task flow at runtime.

Figure 52–6 Calendar Mini View Task Flow



52.2.4 Setting Security for the Events Service

When you create the connection to the Microsoft Exchange Server for the Events service, an external application connection is also created. This external application connection is created to store the user name and password of users in the Microsoft Exchange Server. For more information, see Section 52.2.2, "Setting Up a Connection for the Events Service."

52.3 Using the Events Service REST APIs

WebCenter Portal provides REST APIs to support the Events service. Use the Events service REST APIs to create your own interface for providing access to space events.

> **Note:** The Events service REST APIs are available for space events only. You cannot use REST APIs to work with personal events.

This section describes the REST APIs associated with the Events service. It includes the following subsections:

- Section 52.3.1, "Events Entry Point"

- Section 52.3.2, "Events Resource Type Taxonomy"
- Section 52.3.3, "Security Considerations"
- Section 52.3.4, "Events Resource Types"

52.3.1 Events Entry Point

The entry point for the Events service can be reached only through a space. First you need to navigate to the appropriate space and then find the link element with a `resourceType` of:

```
urn:oracle:webcenter:events:gsEvents
```

The corresponding `href` or `template` element provides the URI entry point. The client sends HTTP requests to this entry point to work with the Events service.

For more information about the Resource Index, see Section 54.5.1, "The Resource Index."

For more information about resource types, see Section 54.5.2.1, "Resource Type."

52.3.2 Events Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resources types, see the appropriate section in Section 52.3.4, "Events Resource Types."

The taxonomy for the Events service is:

```
urn:oracle:webcenter:events:gsEvents
  urn:oracle:webcenter:events:gsEvent
urn:oracle:webcenter:events:gsCategories
  urn:oracle:webcenter:events:gsCategory
```

52.3.3 Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

52.3.4 Events Resource Types

This section provides you with all the information you need to know about each resource type. It includes the following subsections:

- Section 52.3.4.1, "urn:oracle:webcenter:events:gsEvents"
- Section 52.3.4.2, "urn:oracle:webcenter:events:gsEvent"
- Section 52.3.4.3, "urn:oracle:webcenter:events:gsCategories"
- Section 52.3.4.4, "urn:oracle:webcenter:events:gsCategory"

52.3.4.1 urn:oracle:webcenter:events:gsEvents

Use this resource to identify the URI to use to retrieve (`GET`) and create (`POST`) space events. The response from a `GET` operation includes each space event in this collection of events, and each event includes links used to operate on that event. The response from a `POST` operation includes the event that was created in this collection of events and a link to operate on that event.

Navigation Paths to gsEvents

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceIndex
   spaces
      gsEvents
```

Supported Methods for gsEvents

The following methods are supported by this resource:

- `GET`
  - **request - body:** none, **Parameters:** `startIndex`, `itemsPerPage`, `utoken`

    For information about these common parameters, see "Common Request Query Parameters".

    The following additional parameters are available:

    * `startDate`—the date from which to start listing space events
    * `endDate`—the date at which to stop listing space events

    For the start and end dates, use the format `YYYY-MM-DD`, for example `2011-09-01`. You can also specify a time (for example, `2011-09-01T09:00:00`) and a timezone sign (for example, for UTC, `2011-09-01T09:00:00Z`).
  - **response - body:** 0 or more events
- `POST`
  - **request - body:** event
  - **response - body:** event

Resource Types Linked to From gsEvents

Table 52–3 lists the resource types that the client can link to from this resource.

Table 52–3 Related Resource Types for gsEvents

rel	resourceType
self	urn:oracle:webcenter:events:gsEvents

52.3.4.2 urn:oracle:webcenter:events:gsEvent

Use this resource type to identify the URI to use to read (`GET`), update (`PUT`), or delete (`DELETE`) a specific space event. The response from a `GET` operation includes the specific event identified by the URI. The response from a `PUT` operation includes the modified version of the event identified by the URI. The response from a `DELETE` operation is a 204.

Navigation Paths to gsEvent

```
resourceIndex
   spaces
      gsEvents
         gsEvent
```

Supported Methods for gsEvent

The following methods are supported by this resource:

- `GET`
  - **request - body:** none
  - **response - body:** event
- `PUT`
  - **request - body:** event
  - **response - body:** event
- `DELETE`
  - **request - body:** none
  - **response - body:** none

Writable Elements for gsEvent

Table 52–4 lists the writable elements for this resource.

Table 52–4 Writable Elements for gsEvent

Element	Type	Required	Constraints	Description
`category`	String	No	The name of an event category defined for the space	The category to which the event belongs
`details`	String	No	1 or more characters	Additional details about the event
`endTime`	Date	Yes	YYYY-MM-DDT HH:MM:SS `endTime` must be greater than or equal to `startTime`	The date and time at which the event ends
`location`	String	No	1 or more characters	The location at which the event takes place
`priority`	String?	No	1 - Highest 2 - High 3 - Normal 4 - Low 5 - Lowest	The priority of the event, which determines where it appears when events clash
`startTime`	Date	Yes	YYYY-MM-DDT HH:MM:SS	The date and time at which the event starts
`summary`	String	Yes	1 or more characters	A brief description of the event to serve as the title of the event

Read-only Elements for gsEvent

Table 52–5 lists the read-only elements for this resource.

Table 52–5 Read-only Elements for gsEvent

Element	Type	Description
`author`	personReference	User who created the event
`created`	Date	Date on which the event was created
`duration`	String	The length (in minutes) of the event
`groupSpace`	groupSpaceReference	The space to which the event belongs
`id`	String	Unique ID of the event
`isAllDayEvent`	Boolean	Indicates whether the event takes place over an entire day
`modified`	Date	Date on which the event was last updated
`modifiedByUser`	personReference	User who last updated the event

Resource Types Linked to from gsEvent

Table 52–6 lists the resource types that the client can link to from this resource.

Table 52–6 Related Resource Types for gsEvent

rel	resourceType
self	urn:oracle:webcenter:events:gsEvent

52.3.4.3 urn:oracle:webcenter:events:gsCategories

Use this resource to identify the URI to use to retrieve (`GET`) and create (`POST`) space event categories. The response from a `GET` operation includes each category in this collection of categories, and each category includes links used to operate on that category. The response from a `POST` operation includes the category that was created in this collection of categories and a link to operate on that category.

Navigation Paths to gsCategories

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceIndex
   spaces
      gsCategories
```

Supported Methods for gsCategories

The following methods are supported by this resource:

- `GET`
  - **request - body:** none
  - **response - body:** 0 or more categories
- `POST`
  - **request - body:** category
  - **response - body:** category

Resource Types Linked to From gsCategories

Table 52–7 lists the resource types that the client can link to from this resource.

Table 52–7 Related Resource Types for gsCategories

rel	resourceType
self	urn:oracle:webcenter:events:gsCategories

52.3.4.4 urn:oracle:webcenter:events:gsCategory

Use this resource type to identify the URI to use to read (`GET`), update (`PUT`), or delete (`DELETE`) a specific space event category. The response from a `GET` operation includes the specific category identified by the URI. The response from a `PUT` operation includes the modified version of the category identified by the URI. The response from a `DELETE` operation is a 204.

Navigation Paths to gsCategory

This section shows how the client can navigate through the hypermedia to access the gsCategory resource:

```
resourceIndex
   spaces
      gsCategories
         gsCategory
```

Supported Methods for gsCategory

The following methods are supported by this resource type:

- `GET`
  - **request - body:** none
  - **response - body:** category
- `PUT`
  - **request - body:** none
  - **response - body:** category
- `DELETE`
  - **request - body:** none
  - **response - body:** none

Writable Elements for gsCategory

Table 52–8 lists the writable elements for the resource.

Table 52–8 Writable Elements for gsCategory

Element	Type	Required	Constraints	Description
`name`	String	Yes	1 or more characters	The name of the category

Resource Types Linked to from gsCategory

Table 52–9 lists the resource types that the client can link to from this resource.

Table 52–9 Related Resource Types for gsCategory

rel	resourceType
self	urn:oracle:webcenter:events:gsCategory

52.4 Troubleshooting the Events Service

This section provides information to assist you in troubleshooting problems you may encounter while using the Events service.

Problem

Error message displayed in calendar taskflow:

```
Personal Event Service is not configured properly. Rectify the service
configuration and then click Try Again.
```

Solution

Confirm that you have created a Personal Event Connection. For more information, see Section 52.2.2, "Setting Up a Connection for the Events Service."

Problem

After logging into the calendar with the right credentials, you continue to see the **Login to Personal Calendar** link rather than your events.

Solution

Confirm that the page on which calendar taskflow is added is a secured page. Anonymous users cannot log into the Personal Event Calendar server.

53 Integrating the RSS Service

The RSS service enables your application users to add and view RSS 2.0 formatted feeds within your application. This chapter describes how to integrate the RSS Viewer into a Framework application at design time.

This chapter includes the following sections:

- Section 53.1, "Introduction to the RSS Service"
- Section 53.2, "Basic Configuration for the RSS Service"

For more information about managing and including RSS feeds, see:

- "Managing the RSS Service" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*
- "Working with the RSS Service" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*

53.1 Introduction to the RSS Service

Really Simple Syndication (RSS) provides a means of accessing the content of many different web sites from a single location—a news reader. Oracle WebCenter Portal provides the RSS service that encompasses the RSS Viewer and the service to show RSS feeds from various WebCenter Portal services. The RSS Viewer enables you to view external news feeds from different web sites from within your Framework applications.

This section provides an overview of the features and requirements of the RSS service. It includes the following subsections:

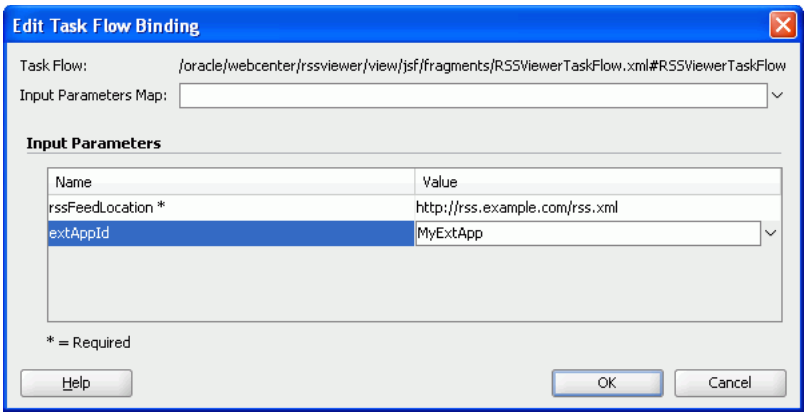
- Section 53.1.1, "Understanding the RSS Service"
- Section 53.1.2, "Requirements for the RSS Service"
- Section 53.1.3, "What Happens at Runtime"

53.1.1 Understanding the RSS Service

To display news feeds from external sources on your application pages, you add the RSS Viewer task flow and specify the URL for the required RSS feed, as shown in Figure 53–1. Your application users can then view the RSS feed at runtime.

For accessing secure application content, the RSS Viewer task flow supports integration with external applications to provide credential mapping services to authenticate with a remote feed. For information about using external applications, see Section 69.13, "Working with External Applications."

Figure 53–1 RSS Details Specified at Design Time



53.1.2 Requirements for the RSS Service

The RSS service does not require any back-end server. You do not need to set up a connection to use this service. However, you can set up a proxy server for the RSS service, if required.

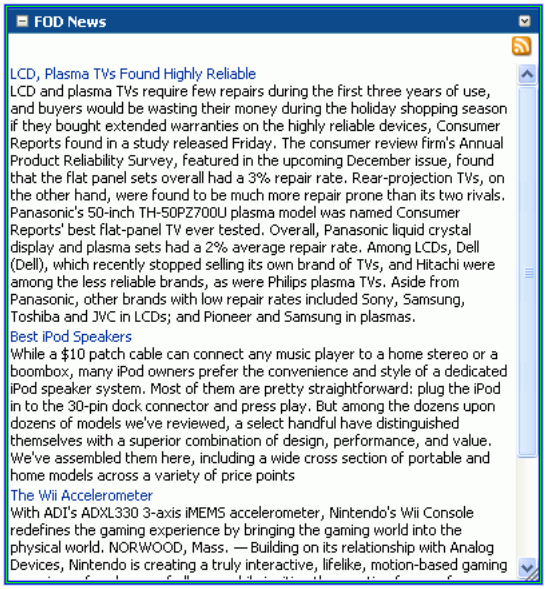
53.1.3 What Happens at Runtime

At runtime, RSS news feeds are displayed from the RSS feed location specified at design time. Users can click the RSS icon on the top-right corner of the application page to open the URL specified as the RSS feed location.

Any user who has permissions to modify the application page can access the RSS Viewer parameters and change the URL of the RSS feed that is rendered.

For information about the RSS service at runtime, see the "Working with the RSS Service" chapter in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Figure 53–2 RSS Feed at Runtime



53.2 Basic Configuration for the RSS Service

This section describes how to set up the proxy server for the RSS service and add the RSS Viewer task flow to your Framework application.

This section includes the following subsections:

- Section 53.2.1, "Setting Up a Proxy Server for the RSS Service"
- Section 53.2.2, "Adding the RSS Service at Design Time"
- Section 53.2.3, "Setting Security for the RSS Viewer Service"

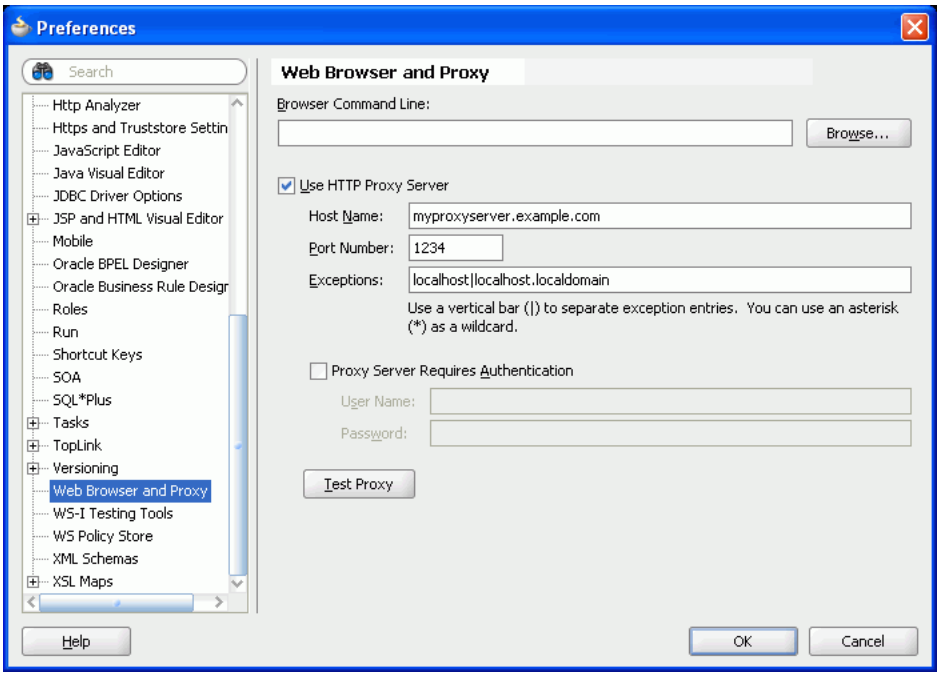
53.2.1 Setting Up a Proxy Server for the RSS Service

The RSS service does not require any connections. You can simply point to the URL of the RSS feed. However, if you want to point to an RSS feed that is external to your intranet and application, you may need to set up a proxy server for your application.

To set up a proxy server for the RSS service:

1. In Oracle JDeveloper, from the **Tools** menu, choose **Preferences**.
2. In the Preferences dialog, scroll down the list on the left side and select **Web Browser and Proxy**.
3. In the right pane, under Web Browser and Proxy, select **Use HTTP Proxy Server** and enter the host name and port number of your proxy server, and note any exceptions (Figure 53–3).
4. Click **OK**.

Figure 53–3 Setting Up a Proxy Server



53.2.2 Adding the RSS Service at Design Time

This section describes the RSS Viewer task flow and how to add it to your application.

This section contains the following subsections:

- Section 53.2.2.1, "About RSS Viewer Task Flow"
- Section 53.2.2.2, "How to Add the RSS Viewer Task Flow to Your Application"
- Section 53.2.2.3, "How to Modify RSS Viewer Task Flow Parameters"

53.2.2.1 About RSS Viewer Task Flow

The RSS service has a single task flow called RSS Viewer, which you can add to your application to enable your users to access an RSS feed. You can add multiple instances of the task flow to your application and use the Edit Task Flow Binding dialog to point to multiple RSS feed locations.

53.2.2.2 How to Add the RSS Viewer Task Flow to Your Application

To add the RSS Viewer task flow to your application:

1. Follow the steps in Section 7.2.1, "How to Prepare Your Application to Consume Services" to implement security, and create a customizable page in your application, if necessary.
2. Open the customizable page on which you want to add the RSS Viewer task flow.
3. If the RSS feed you want to use requires authentication, create an external application. If it does not require authentication, proceed to Step 4.

> **Note:** For more information about external applications, refer to Section 69.13, "Working with External Applications."

4. In the Resource Palette, open the **WebCenter Portal - Services Catalog**, then expand the **Task Flows** folder.
5. Click **RSSViewer**, drag it to your page in the Design view, and choose **Region**.
6. In the Edit Task Flow Binding dialog, specify the URL of the RSS feed. If an external application is used for authenticating the RSS feed, specify its name (the application name, not the application display name), as shown in Figure 53–4.

Figure 53–4 Example of the Edit Task Flow Binding Dialog for the RSS Service

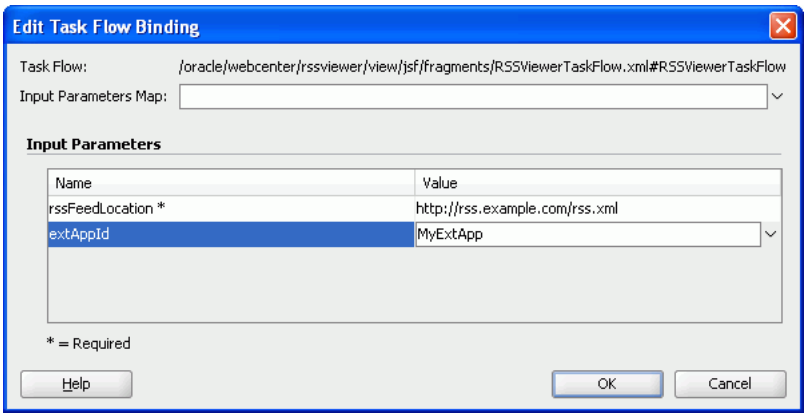


Table 53–1 describes the possible values for the RSS Viewer task flow binding parameters.

Table 53–1 RSS Viewer Task Flow Binding Parameters

Parameter	Value
`rssFeedLocation`	Enter the location of the RSS feed. For example, to use the Oracle Press Releases RSS feed, enter: `${'http://www.oracle.com/rss/rss_ocom_pr.xml'}`
`extAppId`	Enter the name of the external application you want to use to authenticate the Framework application with the RSS feed. If the RSS feed does not require authentication, you do not need to set up and identify an external application for this service.

7. Click **OK** and save your page. The binding displays on your page.

   If you look at the Source tab of your page, you can see the RSS Viewer task flow in the page source, as shown in Example 53–1.

Example 53–1 RSS Viewer Task Flow in the Page Source

```
<af:form id="f1">
     <af:region value="#{bindings.RSSViewerTaskFlow1.regionModel}" id="r1"/>
     <af:region value="#{bindings.RSSViewerTaskFlow2.regionModel}" id="r2"/>
</af:form>
```

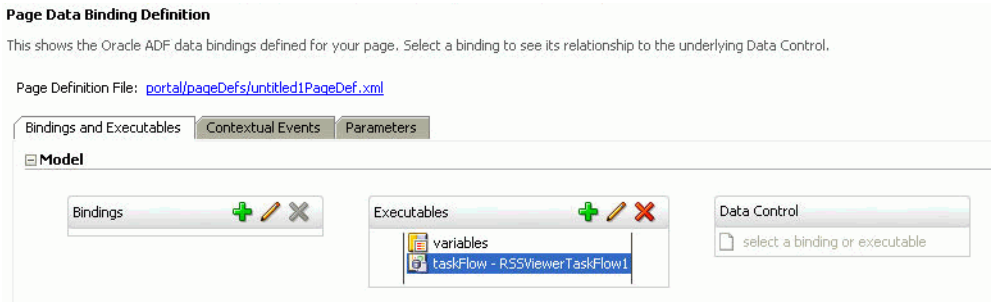
53.2.2.3 How to Modify RSS Viewer Task Flow Parameters

The RSS Viewer task flow has required and optional task flow binding parameters.

You can adjust the parameter values when you drop the task flow onto a page or after you have placed a task flow on a page:

1. Click the **Bindings** tab at the bottom of the page (next to the **Source** tab).
2. Under **Executables**, the RSS Viewer task flow you added is listed (Figure 53–5).

Figure 53–5 Page Data Binding Definition



3. Select the task flow, and next to the Executables heading, click the **Edit selected element** (pencil) icon.
4. In the Edit Task Flow Binding dialog, revise the binding parameter values as required.
5. When you are finished, click **OK.**
6. Save and run your page to see the results.

53.2.3 Setting Security for the RSS Viewer Service

To use the RSS service with a public RSS feed, you do not need to set security. To use the RSS service with an RSS feed that requires authentication, you can set up an external application for your Framework application that sets up either user credentials or public credentials for accessing the RSS feed.

Note: For secure application content, your news reader must support BASIC authentication.

For more information about using external applications, see Section 69.13, "Working with External Applications."

Only authenticated users can view secure RSS feeds. If a user is not authenticated and the RSS feed is secured, the user will not see any content in the RSS Viewer unless the external application specifies public credentials.

Note: When you add the RSS Viewer task flow to your Framework application, the View grant is automatically added to the `authenticated-role`.

Part IX

Extending Your WebCenter Portal Application

Part IX contains the following chapters:

- Chapter 54, "Using Oracle WebCenter Portal REST APIs"
- Chapter 55, "Integrating Other Oracle Applications"
- Chapter 56, "Extending the Spaces Application Using JDeveloper"
- Chapter 57, "Integrating with Oracle WebCenter Portal: Spaces"

54

Using Oracle WebCenter Portal REST APIs

Oracle WebCenter Portal provides a set of REST APIs for retrieving and modifying server data. This chapter discusses the WebCenter Portal REST APIs.

This chapter includes the following sections:

- Section 54.1, "Introduction to REST"
- Section 54.2, "Understanding the Username-Based Security Token Encryption"
- Section 54.3, "Benefits of Using REST"
- Section 54.4, "Introduction to WebCenter Portal's REST APIs"
- Section 54.5, "Understanding the Link Model"
- Section 54.6, "Understanding Items Hypermedia"
- Section 54.7, "Navigating Hypermedia Using HTTP"
- Section 54.8, "Security Considerations for WebCenter Portal REST APIs"
- Section 54.9, "Security Considerations for CMIS REST APIs"
- Section 54.10, "Understanding Common Types"
- Section 54.11, "Managing Caches"
- Section 54.12, "Configuring a Proxy Server"
- Section 54.13, "WebCenter Portal's REST API Examples"



This chapter includes some examples that demonstrate how to use WebCenter Portal REST APIs. For more examples, see the Oracle WebCenter Portal Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

`http://www.oracle.com/technology/products/webcenter/release11_demos.html`

54.1 Introduction to REST

REST (REpresentational State Transfer) is an architectural style for making distributed resources available through a uniform interface that includes uniform resource identifiers (URIs), well-defined operations, hypermedia links, and a constrained set of media types. Typically, these operations include reading, writing, editing, and removing, and media types include JSON and XML/ATOM.

REST commands use standard HTTP methods as requests to point to the resource being used. Every request returns a response, indicating the status of the operation. If the request results in an object being retrieved, created, or updated, the response includes a standard representation of that object.

REST supports multiple clients, both from client machines and other servers, and it can be used from just about any client or development technology, including Java, JavaScript, Ruby on Rails, PHP, .Net, and so on.

> **Tip:** For a good general introduction to REST, see the Wikipedia article Representational State Transfer at `http://en.wikipedia.org/wiki/Representational_State_Transfer`.

REST is typically used with Rich Internet Applications (RIA) that are client-side scripted and require the ability to interact with data from a server-side application. For example, the WebCenter iPhone App uses WebCenter Portal REST APIs to interact with a WebCenter Portal: Spaces application. The native iPhone client is written in Objective-C, and the REST APIs enable the client to send and retrieve application data.

54.2 Understanding the Username-Based Security Token Encryption

To provide additional security, every URI, for both `href` and `template` attributes, includes a security token parameter that is based on the authenticated username (a utoken).

The security generation algorithm uses a randomly generated "salt" along with the username. The salt ensures that if any of the parameters used to perform encryption become compromised, existing tokens can be invalidated and new ones generated. Most importantly, because the username is used as part of the user token generation algorithm, the salt prevents having to change all user names in the event of a compromise.

The salt is stored in the Credential Store Framework (CSF) in the map `o.webcenter.jf.csf.map` against key `user.token.salt`. You can change the value of this key (and other keys used for token encryption) by accessing CSF in Oracle Enterprise Manager.

> **Caution:** If you change the encryption key values, all existing username based security tokens will immediately become invalid. Only change these values under extraordinary circumstances, like an algorithm parameter compromise.

54.3 Benefits of Using REST

Many excellent articles have been published about REST and the benefits of the RESTful style of software architecture. Some of these benefits include:

- Using "Hypermedia As The Engine Of Application State" (HATEOAS) links to access the REST API helps promote API robustness. Since the clients only use URIs returned from the server, if the server changes the URI format, the client will continue to function properly. See also Section 54.5, "Understanding the Link Model."
- Using the standard HTTP protocol allows network infrastructure to cache REST requests where appropriate, reducing load on both the client and server. See also Section 54.11, "Managing Caches."
- Stateless REST requests allow each request to be served by any number of different servers, helping with scalability.

- Using the standard HTTP protocol allows a wide variety of clients to interact with the REST APIs without requiring specialized libraries.

54.4 Introduction to WebCenter Portal's REST APIs

In addition to enabling mobile access, WebCenter Portal REST APIs allow you to take advantage of Web 2.0 technologies like Ajax, JavaScript, and JSON to create rich, interactive browser-based user interfaces and to access and modify Oracle WebCenter Portal: Framework and WebCenter Portal: Spaces data. In general, the WebCenter REST commands provide a more natural and easy-to-use alternative to a SOAP-style web services approach.

Table 54–1 describes the Oracle WebCenter Portal REST APIs provided for WebCenter Portal: Services features.

Table 54–1 Summary WebCenter Portal: Services Features Supported by REST APIs

REST API	Description	Chapter
Discussions	Enable a client to post, read, update, and delete discussion forums, topics, and messages.	Section 34.3.8, "Using the Discussions Service REST APIs."
Lists	Enables a client to browse all the lists associated with a named group space; search list columns given a search term; create new lists; add, update, and remove list rows; and similar sorts of list-related tasks	Section 48.4, "Using the Lists Service REST APIs."
People Connections	Enable a client to view profile data; manage connection lists, feedback, and messages; create new activities and view activities for users, lists, and group spaces.	Chapter 42, "People Connections Service REST APIs."
WebCenter Portal: Spaces	Enable a client to retrieve group space metadata and view, create, update, and delete group space lists and list items. You can also retrieve group space membership information.	Section 57.3, "Using the WebCenter Portal: Spaces REST APIs."
Content Management	Uses the CMIS (Content Management Interoperability Services) RESTful server binding to provide access to the CM VCR (Content Management Virtual Content Repository).	*Oracle Fusion Middleware Content Management REST Service Developer's Guide*.
Activity Graph	Enables you to retrieve recommendations for connections, group spaces, and items using the underlying Activity Graph engine.	Section 46.3.2, "Using the Activity Graph Service REST APIs."
Events	Lets you access calendar events associated with a named group space.	Section 52.3, "Using the Events Service REST APIs."
Feedback	Enables a client to create, read, and delete feedback in a social networking application.	Chapter 42.3, "Feedback REST APIs."
Search	Lets you post, read, update, and delete searches. You can specify keywords and the scope of the search; for example, the iPhone could search for "smith" in all spaces, documents and wiki pages.	Section 45.3.7, "Using the Search Service REST APIs."
Tags	Enables a client to read, post, update, and delete tags and tagged items.	Section 44.3.2, "Using the Tags Service REST APIs."

Table 54–1 (Cont.) Summary WebCenter Portal: Services Features Supported by REST

REST API	Description	Chapter
Navigation	Use the navigation REST APIs to create your own interface for displaying navigations. **Note:** The navigation REST APIs do not share the resource index described in this chapter.	Section 14.3.2, "Using the Navigation REST APIs."
Personalization	Enables you to access the Personalization Conductor.	Section 67.4, "Extending Personalization."
Pagelets	Allows remote web services to retrieve information about resources and pagelets. Inject pagelets into non-proxied pages, allowing the Pagelet Producer to act as a portlet provider for Oracle WebCenter Interaction, Oracle WebLogic Portal, or other third-party portals.	Section 63.2.2.2, "Accessing Pagelets Using REST."

54.5 Understanding the Link Model

Hypermedia is at the core of two of the most successful web-based formats: HTML and ATOM. HTML and ATOM allow consumers to navigate to other hypermedia documents through links–for example, clicking on a link to go to a news article.

Hypermedia drives RESTful application state (known as HATEOAS: Hypermedia As The Engine Of Application State).

Note: HATEOAS analogy to define application state:

Suppose you are completing your taxes in your favorite browser. You finish entering your W-2 data and move on to deductions when the browser crashes. The state you lost—the fact that you were on deductions and still needed to enter data—is the application state; not the W-2 data entered (that is, change states from the current state).

HATEOAS dictates that *this* state—the application state—be captured wholly in hypermedia. Application state is where you are in the application, not what data you've entered into the application. One of the benefits of this approach is that it simplifies the client and server, because they do not need to be aware of the state they are in. The link contains all the state information necessary to process the request, so, when the browser restarts and returns to the link, the user will be at the same place in the tax process.

Given a set of top-level URI entry points to a RESTful service, all interactions beyond those entry points are driven by hypermedia links returned in response representations. This link-centered approach helps keep the client from becoming too tightly coupled to the server URLs. The client is using URLs given to it by the server, therefore the client code does not break if the server URLs change format.

Understanding this link model helps you understand how to use the data the service returns to navigate the REST APIs.

This section describes the hypermedia link model used by WebCenter's RESTful services. It includes the following subsections:

- Section 54.5.1, "The Resource Index"
- Section 54.5.2, "The Anatomy of a Link"

54.5.1 The Resource Index

In WebCenter, the *Resource Index* is your starting point for all authenticated access. The Resource Index provides access to the set of top-level URI entry points. It provides the way in to all the available WebCenter RESTful services. The Resource Index URI is the only URI that you need to know.

The WebCenter Resource Index URI is:

`http://host:port/rest/api/resourceIndex`

> **Note:** Access to the Resource Index always requires authentication; however, you can (optionally) access the CMIS resource entry point anonymously using the following URI:
>
> `http://host:port/rest/api/cmis/repository`
>
> See also Section 54.9, "Security Considerations for CMIS REST APIs."

The first step in using the WebCenter Portal REST APIs is to send a `GET` request to the Resource Index. The response varies depending on the services available and the media type of the request. Example 54–1 shows how the response might look if you made an Ajax request using JavaScript (and possibly a client-side scripting library, such as Dojo) to retrieve the JSON data for the Resource Index. Note that this is an abridged sample response and does not include all of the links actually present in a real response.

Example 54–1 Response to a GET on the Resource Index

```
{
  "resourceType": "urn:oracle:webcenter:resourceindex",
  "links": [
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:messageBoard",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "resourceType": "urn:oracle:webCenter:cmis",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:discussions:forums",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "resourceType": "urn:oracle:webcenter:resourceindex",
      "rel": "self",
      "href": "http://host:port/rest/api/resourceIndex",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:activities:stream",
```

```
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:people:person",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:feedback",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:spaces",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    },
    {
      "template": "opaque-template-uri",
      "resourceType": "urn:oracle:webcenter:people",
      "href": "opaque-uri",
      "capabilities": "urn:oracle:webcenter:read"
    }
  ]
}
```

By interpreting the links returned in the Resource Index data, you can retrieve the URI entry point for an individual service by locating the URI for the resource type you want to use. You can then continue navigating through the hypermedia until you can perform the required operation. Example 54–2 shows a method that locates a URI given the Resource Index JSON data.

Example 54–2 Locating the URI for a Particular Service in the Resource Index

```
/* Parse the resourceIndex to find the specified URL and
 * return it.
 *
 * @Param jsonData the JSON data retrieved from calling
 *        the /rest/api/resourceIndex URL.
 * @Param strResourceType the resource type of the URL
 *        you want to retrieve from the resourceIndex data.
 *        E.g., 'urn:oracle:webcenter:activities:stream'
 */
function getResourceURL(jsonData, strResourceType)
{
  // Using the HATEOAS model, we browse the returned links
  // looking for the one with the correct resource type.
  for (var i = 0; i < data.links.length; i++) {
    if (data.links[i].resourceType == strResourceType) {
      return data.links[i].href;
    }
  }
}
```

54.5.2 The Anatomy of a Link

Example 54–3 and Example 54–4 show the anatomy of a hypermedia link as XML and in JSON document fragments, respectively.

Example 54–3 Link in an XML Document Fragment

```
<links>
  <link href="opaque-URI"
        template="opaque-template-URI (optional)"
        rel="rel-name"
        title="human-readable-title (optional)"
        type="media-type (optional)"
        resourceType="resource-type"
        capabilities="operation"/>
  ...repeat as needed...
</links>
```

Example 54–4 Link in a JSON Document Fragment

```
"links": [
  {
    "href":"opaque-URI",
    "template":"opaque-template-URI (optional)",
    "rel":"rel-name",
    "title":"human-readable-title (optional)",
    "type":"media-type (optional)",
    "resourceType":"resource-type",
    "capabilities":"operation"
  },
  ...repeat as needed...
]
```

The `resourceType`, `rel`, and `capabilities` attributes of the hypermedia link provide metadata that enable clients to determine which URI (`href` or `template`) to use, without having to parse the URIs directly. The URIs are opaque—the metadata determines which link is useful in a given circumstance.

Multiword field, element, and attribute names are formatted in camel case, unless the representation is attempting to conform to a specification not under the service author's direct control. Acronyms are treated as normal words with their case adjusted accordingly (for example, `fooXml` or `xmlFoo`). See Example 54–5 and Example 54–6.

Example 54–5 XML Naming Convention

```
<myElement>text</myElement>
```

Example 54–6 JSON Naming Convention

```
{"myElement": "text"}
```

This section includes the following subsections that describe the different attributes of the hypermedia link:

- Section 54.5.2.1, "Resource Type"
- Section 54.5.2.2, "Relationship"
- Section 54.5.2.3, "Capabilities"
- Section 54.5.2.4, "Media Type"

- Section 54.5.2.5, "Templates"

54.5.2.1 Resource Type

The `resourceType` link attribute indicates the type of resource to which the link points. Clients should use the `resourceType` to determine the expected response bodies for `GET` and `POST` and allowable request bodies for `POST` and `PUT`.

For more information, see Section 54.7, "Navigating Hypermedia Using HTTP."

54.5.2.2 Relationship

The `rel` link attribute indicates the relationship of the linked object to the current object (that is, the object that contains the list of links). The value of this attribute is a space-separated list of the following currently supported values:

- `self`—The linked object is the current object
- `related`—The linked object is related to the current object
- `via`—The linked object is the source of the information for the current object
- `alternate`—The linked object is a substitute for the current object (typically, the same object in another format, such as an HTML page that displays the current object)
- `urn:oracle:webcenter:parent`—The linked object is the parent of the current object. That is, the linked object owns the current object

> **Note:** Some REST APIs for some WebCenter features may contain additional `rel` link attributes. See the REST API documentation for each specific feature for more information.

54.5.2.3 Capabilities

The `capabilities` link attribute indicates which methods are supported by the linked resource.

Links are returned only if a client is allowed to access that resource. User authorization can affect the capabilities a client has with the links returned in a response representation. In general, services only return the capabilities that the current authorized user has permission to execute and that the resource supports.

If there is no link, then the client cannot access the resource. If a link has no capabilities, then it is not returned to the client, meaning that the client does not have permission to do anything with that link (even read it).

Capability-based expression of hypermedia links communicates the range of operations that the client can expect to succeed, which allows the client to dynamically configure any associated UI to provide the best overall user experience.

The value of this attribute is a space-separated list of the following values:

- `urn:oracle:webcenter:create`—This maps to the HTTP verb `POST`
- `urn:oracle:webcenter:read`—This maps to the HTTP verb `GET`
- `urn:oracle:webcenter:update`—This maps to the HTTP verb `PUT`
- `urn:oracle:webcenter:delete`—This maps to the HTTP verb `DELETE`

Note: The top-level `resourceIndex` links only returns the `read` capability, even if the user is authorized with additional capabilities.

Note: Querying a resource for the allowed HTTP verbs using `OPTIONS` returns the verbs that the resource can support in general, and does not take a user's access into account. The capabilities attribute in a link describes exactly what the current user can do with the current resource. `OPTIONS` may return more HTTP verbs than the current user is allowed.

54.5.2.4 Media Type

The `type` link attribute indicates the media types supported by the linked object.

All REST services, except for CMIS, support both XML (`application/xml`) and JSON (`application/json`) media type. CMIS currently supports only ATOM. For more information about CMIS REST APIs, see the *Oracle Fusion Middleware Content Management REST Service Developer's Guide*.

54.5.2.5 Templates

The `template` link attribute indicates that the client can use a URI template, instead of the `href` URI, to provide parameterized values for the linked object. Links must include at least an `href` or a `template` URI, but can include both.

Some hypermedia links support request query parameters that allow the client to configure the link in different ways. Rather than force the client to know the URI format and manually build the URI, URI templates are used. These templates allow client code to easily insert data into a URI without having to understand exactly how the URI works. This maintains the opacity of hypermedia URIs and protects the client from changes to the URI format.

Example 54–7 shows a URI template including several request query parameters.

Example 54–7 URI Template

```
http://host:port/.../lists?startIndex={startIndex}&itemsPerPage={itemsPerPage}&q={
searchTerms}&projection={projection]
```

WebCenter Portal REST APIs use a simple slot replacement syntax that follows many industry URI template schemes.

For example, using the template in Example 54–7, to see 10 list items on the first page, the client would provide a value of `1` for the `startIndex` parameter and a value of `10` for the `itemsPerPage` parameter, as shown in Example 54–8.

Example 54–8 URI Template with Parameter Values

```
http://host:port.../lists?startIndex=1&itemsPerPage=10
```

> **Note:** All unused parameters must be removed from a template before it can be used. Clients may not submit unprocessed templates to the service that produced it; doing so results in undefined behavior, generally returning a status code of 500.
>
> Clients must process templates into valid URI form before submitting to the server. Clients must replace slots with appropriate values, taking care to properly URI encode any value replacing the slot token. If a client does not have a suitable value for one or more of the slots in the template, then it must replace the slot token with an empty string.

You must URL-encode special or reserved character in parameter values. For example, to search lists for a person named Günter, you must URL-encode the ü as shown in Example 54–9.

Example 54–9 Encoding Special Characters in URI Templates

```
http://host:port.../lists?q=G%FCnter
```

Common Request Query Parameters

Many resources support a common set of request query parameters. For example, when retrieving a collection of entities, it is common to change the shape of the results set by limiting the quantity or details of the results. The REST framework uses the following request parameters to scope results and provide security:

- `startIndex`—Specifies the index of the first matching result that should be included in the result set (0-*n* ... zero based). This is used for pagination.
- `itemsPerPage`—Specifies the maximum number of results to return in the response (1-*n*). This is used for pagination.
- `q`—Specifies implementation-specific searching. Searches may be specified using the following format (square brackets [] denote optional values):

  ```
  [[field1:[operand]][:]value1[;field2:operand:value2]]
  ```

  For example:

  ```
  &q=login:equals:monty
  &q=title:contains:issues
  &q=creator:equals:monty;description:contains:Urgent
  ```

  While each resource uses the same format for the `q` parameter, the way search is implemented is different depending on the resource being searched. For more information about how each resource implements search, see the chapter for the specific service.

- `projection`—Reserved for implementation-specific projection of model representations, such as variable recursion depth, field or attribute filtering. Valid values are `summary` or `details`.

  For example, requesting a projection of `summary` for a collection of lists, returns only the title, description, and hypermedia links. Requesting a projection of `details` results in the server sending back a collection of lists that includes all the column metadata for each list. This may require additional processing time or database queries on the server.

The following example request results in the response entity containing a deeper object graph.

```
http://host:port/...lists&projection=details
```

- `data`—This parameter accepts a comma separated list of data sets and items. This parameter lets clients specify what data they would like to receive. For example, a mobile device application might use this parameter to limit the amount of XML data returned. If both the `projection` and `data` query string parameters are present, the `data` parameter will be used to determine which data to return. If you specify the constant `'data'` as the data parameter, all the standard information will be returned for the resource.

For information about how these parameters are supported by specific resources, see the chapter for the appropriate service.

54.6 Understanding Items Hypermedia

A collection of `items` makes up the actual content of responses. This is at the same level as the `links` section described previously. Each item (including the top-level tag in each response) has one common attribute (`resourceType`) in addition to resource-specific content and format. For details beyond the `resourceType`, see the chapter for the specific service.

See Example 54–13 for an example that describes a collection of entities/items. Example 54–15 describes a single entity/item.

54.7 Navigating Hypermedia Using HTTP

You can navigate REST service hypermedia in a similar way to that used to browse and interact with HTML or an ATOM feed. Interactions are performed on the resources identified by links using HTTP methods. The REST services return response codes and response bodies to the client, and the client uses the hypermedia in the response to drive further interactions.

Table 54–2 describes the general pattern followed when constructing opaque resource URIs. The `resourceType` differentiates whether the HTTP method operates on a collection of resources or an individual resource.

Table 54–2 HTTP Methods

HTTP Method	Response for a Collection of Resources	Response for an Individual Resource
`GET`	Returns resource collection container (200 HTTP response code)	Returns resource (200 HTTP response code)
`PUT`	Cannot update a collection of resources (405 HTTP response code)	Updates and returns resource (200 HTTP response code)
`POST`	Creates and returns a new resource within the collection. **Note:** Can return a 201 or 204 HTTP response code. The returned code depends on whether the newly created object is directly addressable or not. For instance, activities cannot be addressed individually, so they return a 204 no-content response code.	Cannot create a resource within an individual resource (405 HTTP response code)

Table 54–2 (Cont.) HTTP Methods

HTTP Method	Response for a Collection of Resources	Response for an Individual Resource
DELETE	Cannot delete a collection of resources (405 HTTP response code)	Deletes resource (204 HTTP response code)

Collection resources generally support reading a collection (GET) and creating a subordinate to that collection (POST). Individual resources generally support reading a resource (GET), updating a resource (PUT), and deleting a resource (DELETE).

HTTP Response Status Codes

Table 54–3 describes the potential response status codes.

Table 54–3 HTTP Response Status Codes

HTTP Response Status Code	Description
200	OK. Upon successful completion of a GET or PUT. This is accompanied by a resource representation as a response body.
201	Created. Upon successful completion of a POST. This has a location header to the newly-created resource.
204	No content, or any request that does not return content. For example, creating an object that cannot be linked. Upon successful completion of a DELETE.
400	Bad Request. The URI was malformed or could not be processed; for example, the IDs were not formatted correctly, or the ID was supplied in the URI on a POST.
401	Unauthorized. Client may retry by submitting credentials. This may be accompanied with a fault response body to help diagnose the issue.
403	Forbidden. Client does not have permission to perform a particular action, such as creating or deleting a resource. Re-authenticating as the same user does not help. This may be accompanied with a fault response body to help diagnose the issue.
404	Not Found. Referencing a specific resource with an ID, but that resource does not exist.
405	Method Not Allowed. This includes a list of valid methods for the requested resource.
406	Not Acceptable. The Accept header media type(s) sent by the client are not supported for the requested operation.This may be accompanied with a fault response body to help diagnose the issue.
409	Conflict. Possibly the resource ID is in use, or an entity has been modified by another process during an update.This may be accompanied with a fault response body to help diagnose the issue.
422	Bad entity body, the data in the body, although syntactically correct, was not valid, or could not be processed; for example, invalid data when updating a row.
500	Internal server error. The server encountered an unexpected condition that prevented it from fulfilling the request.
501	Not Implemented. The server does not support the functionality required to fulfill the request. This is the appropriate response when the server does not recognize the request method and is not capable of supporting it for any resource.

54.8 Security Considerations for WebCenter Portal REST APIs

All of the WebCenter REST URIs reference protected resources (similar to protected web pages) and require authentication for access.

> **Note:** The one exception to the authenticated access rule is access to the CMIS resources. CMIS resources can be accessed anonymously through the CMIS URI entry point:
>
> `http://host/port/rest/api/cmis/repository`
>
> See also Section 54.9, "Security Considerations for CMIS REST APIs."

You can pass this authentication in with the request using basic authentication, or you can configure the client and the WebCenter REST service to use single sign-on. For more information about single sign-on, see the section "Configuring a WebCenter Portal: Framework Application to Use Single Sign-On" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Basic authentication sends the user's password in plain text. If you use this type of authentication, you should consider securing the connection using SSL. For more information, see the section "Configuring WebCenter Portal: Framework Applications and Components to Use SSL" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To provide additional security, every URI, for both `href` and `template` attributes, includes a security token parameter. The security token is user-scoped. This means that it is based on and scoped to an authenticated user and can be bookmarked or cached across that user's sessions. These security tokens help prevent Cross-Site Request Forgery (CSRF) attacks.

For example:

```
<link
  template="opaque-template-uri/@me?startIndex={startIndex}&itemsPerPage={itemsPerPage}
     &token=generated-token" resourceType="urn:oracle:webcenter:messageBoard"
     href="opaque-uri/@me?token=generated-token" capabilities="urn:oracle:webcenter:read"
/>
```

> **Note:** The security token is not used for authentication or identity propagation.

WebCenter Portal REST APIs operate under the identity of the authenticated user. For example, the group space REST APIs only return information for, and allow changes to, group spaces to which the user has access.

54.9 Security Considerations for CMIS REST APIs

The CMIS REST APIs do not use the same authentication scheme as the other WebCenter Portal REST APIs. Whereas other WebCenter Portal REST APIs do not allow unauthenticated access and prompt the user for authentication before allowing access, the CMIS REST APIs do allow unauthenticated access.

If a document requires authentication information and does not receive that information (because it is being accessed by an unauthenticated user), a 404 error is returned. This does not necessarily mean that the document cannot be found, rather

that the (unauthenticated) user does not have the appropriate permissions to access the document. For the request to succeed, it should include basic authentication headers to identify the current user.

CMIS stands for Content Management Interoperability Services, a standard REST interface for Enterprise Content Management Systems. For more information, see *Oracle Fusion Middleware Content Management REST Service Developer's Guide*.

54.10 Understanding Common Types

This section describes the common types that are shared by multiple WebCenter Portal REST APIs.

- Section 54.10.1, "Common Types"
- Section 54.10.2, "Portable Contact Types"

54.10.1 Common Types

Common types provide a consistent way to reference objects used in the WebCenter Portal REST APIs.

This section includes the following subsections:

- Section 54.10.1.1, "personReference"
- Section 54.10.1.2, "groupSpaceReference"

54.10.1.1 personReference

This is a generic data type that represents a user in the system. It is used by several APIs, for example to identify the author of a message board or feedback message, or a user's manager or direct reports. It is made up of the following elements:

- `guid`—The GUID of the user
- `id`—The login ID of the user
- `displayName`—The display name of the user

The `personReference` also includes a link to the user's profile icon. You can control the size of the icon by providing values: `small`, `medium`, or `large`.

Depending on where the `personReference` type is included, it can also return links to the associated REST APIs of the generating response. For example, if the `personReference` type is included as the author in a message board response, it includes links to message board services.

54.10.1.2 groupSpaceReference

This is a generic data type that represents a group space. It is used by several APIs, for example in the activity stream, to identify a group space in which a particular activity occurred. It is made up on the following elements:

- `guid`—The GUID of the group space
- `name`—The name of the group space
- `displayName`—The display name of the group space

The `groupSpaceReference` also includes an html link, rest link, and icon link.

54.10.2 Portable Contact Types

Portable Contact types provide users with a standard way to access their address books and friends lists over the web. Portable Contact types are used by the Profile component of the People Connections service.

> **Note:** These types are based on the Portal Contact Types in WebCenter. They may include additional data.

This section includes the following subsections:

- Section 54.10.2.1, "name Portable Contact Type"
- Section 54.10.2.2, "address Portable Contact Type"
- Section 54.10.2.3, "organization Portable Contact Type"
- Section 54.10.2.4, "value Portable Contact Type"

54.10.2.1 name Portable Contact Type

This is a portable contact type that provides information about the user's name. It is made up of the following elements:

- `formatted`—The formatted version of the full name of the user, for example, Michael David Jones Ph.D.
- `familyName`—The family name, or last name, of the user, for example Jones
- `givenName`—The given name, or first name, of the user, for example Michael
- `honorificSuffix`—The honorific suffix of the user, for example, Esq. or Ph.D.
- `initials`—The first initials of the user, for example, M. D.
- `maidenName`—The maiden name of the user

Some of the elements may not be present depending on the user repository configuration and data.

54.10.2.2 address Portable Contact Type

This is a portable contact type that provides information about the user's address. It is made up of the following elements:

- `formatted`—The formatted version of the full address
- `type`—The type of the address, for example, Home, Work
- `streetAddress`—The street address
- `poBox`—The post office box number
- `locality`—The city or locality
- `region`—The state or region
- `postalCode`—The zip code or postal code
- `country`—The country

Some of the elements may not be present depending on the user repository configuration and data.

54.10.2.3 organization Portable Contact Type

This is a portable contact type that provides information about the user's organizational affiliation. It is made up of:

- `name`—The name of the organization
- `employeeNumber`—The employee number of the user
- `employeeType`—The employee type of the user.
- `department`—The department within the organization to which the user belongs
- `defaultGroup`—The default group to which the user belongs
- `title`—The job title of the user within the organization
- `description`—A textual description of the user's role within the organization
- `expertise`—The expertise of the user within the organization
- `startDate`—The date when the user joined the organization

Some of the elements may not be present depending on the user repository configuration and data.

54.10.2.4 value Portable Contact Type

This is a generic object that contains data for a wide variety of contact information. It is made up of the following elements:

- `primary`—A boolean value that identifies whether this is the primary piece of information of this type for this person. The primary element may not be present and is only relevant if there are multiple values for the same type of data.
- `value`—The value for this type
- `type`—The type of information. Valid types are:
  - `standard`: with valid values of `work`, `home`, `other`
  - `phoneNumber`: with valid values of `work`, `home`, `fax`, `pager`, `mobile`
  - `photos`: with a valid value of `thumbnail`

54.11 Managing Caches

Client-side developers need to know how to handle HTTP cache headers in both requests and responses. Individual resources that have a "last modified" date are also return entity tags. The entity tags can be used to make retrieval of a specific entity more efficient. To learn more about the use of entity tags in caching, please see the Hypertext Transfer Protocol specification.

54.12 Configuring a Proxy Server

A proxy server is typically employed to avoid cross-domain request problems associated with making XMLHttpRequest (XHR) calls from a browser client. These calls are typically associated with the Ajax development technique for creating rich, interactive client-side interfaces. REST APIs are typically used within this kind of client-side development scenario.

For more information on setting up a proxy server, see "Configuring a Proxy Server" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

54.13 WebCenter Portal's REST API Examples

This section includes some examples illustrating how to use the WebCenter Portal REST APIs. It includes the following subsections:

- Section 54.13.1, "Navigating the Message Board Hypermedia"
- Section 54.13.2, "Displaying Activity Stream Data"
- Section 54.13.3, "Using the WebCenter People Connections Service REST APIs with Ext"
- Section 54.13.4, "Updating User Status"

54.13.1 Navigating the Message Board Hypermedia

This section includes examples to illustrate how to navigate the REST service hypermedia. The examples show how to read messages on a message board, post messages to another user's message board, and delete unwanted messages.

This section includes the following subsections:

- Section 54.13.1.1, "Accessing the Resource Index"
- Section 54.13.1.2, "Reading Messages"
- Section 54.13.1.3, "Creating a New Message"
- Section 54.13.1.4, "Updating a Message"
- Section 54.13.1.5, "Deleting a Message"

54.13.1.1 Accessing the Resource Index

The first step is always to access the Resource Index (Example 54–10).

Example 54–10 Accessing the Resource Index

```
GET /resourceIndex
```

This request returns a list of the top-level URI entry points to the RESTful services, including the entry point for the message board (Example 54–11).

Example 54–11 Response to Accessing the Resource Index

```
200 OK
Accept: application/json;charset=UTF-8

{
  "resourceType": "urn:oracle:webcenter:resourceindex",
  "links": [
    {
      "resourceType": "urn:oracle:webcenter:resourceindex",
      "capabilities": "urn:oracle:webcenter:read",
      "rel": "self",
      "href": "http://host:port/rest/api/resourceIndex"
    },
    {
      "resourceType": "urn:oracle:webcenter:messageBoard",
      "capabilities": "urn:oracle:webcenter:read",
      "href": "opaque-messageBoard-URI"
    },
  ...repeating for other services...
}
```

You can examine this list to find the URI that you require to access your message board. You should look for the link with a `resourceType` of `urn:oracle:webcenter:messageBoard`. The `href` for this link is the one that you require to access your message board.

For other resources `rel`, `type`, and `template` also help find the correct link.

54.13.1.2 Reading Messages

Once you have determined the correct URI for your message board, you can send a GET request to that URI to read your messages (Example 54–12).

To read messages on a message board, you must be logged in.

Example 54–12 Retrieving Messages from Your Message Board (GET)

```
GET /opaque-messageBoard-URI
```

The response provides information about all the messages on your message board (Example 54–13).

Example 54–13 Response to Retrieving Messages from Your Message Board

```
200 OK
Accept: application/json;charset=UTF-8

{
  "resourceType": "urn:oracle:webcenter:messageBoard",
  "links": [
    {
      "resourceType": "urn:oracle:webcenter:messageBoard",
      "capabilities": "urn:oracle:webcenter:read urn:oracle:webcenter:create",
      "rel": "self",
      "href": "opaque-messageBoard-URI"
    }
  ]
  "items": [
    {
      "resourceType": "urn:oracle:webcenter:messageBoard:message",
      "links": [
        {
          "resourceType": "urn:oracle:webcenter:messageBoard:message",
          "capabilities": "urn:oracle:webcenter:read urn:oracle:webcenter:delete",
          "rel": "self",
          "href": "opaque-message-URI"
        }
      ]
      "id": "89add57c-7a35-4d35-b24f-ea9259612eb8",
      "body": "What's up?  It's been a while.  Some of us are going to Conner
O'Neal's after work.  Want to go?",
      "created": "2009-09-10T11:18:46.696-0700",
      "author": {
        "id": "carl",
        "displayName": "carl",
        "guid": "649A27F09D5C11DEBFAA799CBD41D9B8",
        "links": [
          {
            "resourceType": "urn:oracle:webcenter:people:person",
            "capabilities": "urn:oracle:webcenter:read",
            "rel": "via",
```

```
            "href": "opaque-person-URI"
          },
          {
            "resourceType": "urn:oracle:webcenter:messageBoard",
            "capabilities": "urn:oracle:webcenter:read
urn:oracle:webcenter:create",
            "href": "opaque-messageBoard-URI-for-Carl"
          },
          {
            "type": "text/html",
            "resourceType": "urn:oracle:webcenter:spaces:profile",
            "capabilities": "urn:oracle:webcenter:read",
            "rel": "alternate",
            "href": "opaque-profile-URI"
          }
        ]
      },
    }
  ],
  "startIndex": 0,
  "itemsPerPage": 1,
}
```

From the response you can see that you have `read` and `create` capabilities on your message board. So you can read its contents and post new messages.

In addition, the response also includes a collection of items (in this case the collection consists of just a single item). These items, with a `resourceType` of `urn:oracle:webcenter:messageBoard:message`, are the messages on your message board. The `capabilities` attribute for the message indicates that, for this particular message, you can read it or delete it from your message board.

For each message, the response provides the following information:

- `id`—the identifier of the message
- `body`—the text of the message
- `author`—the author of the message. The author element is also made up of several other elements:
  - `id`—the identifier, or user name, of the author of the message
  - `displayName`—the name of the author, formatted for display
  - `guid`—the globally unique identifier of the author

Within the `author` element there is also a collection of three links. The `resourceType` of these links are:

- `urn:oracle:webcenter:people:person`—enables you to view information about the author of the message
- `urn:oracle:webcenter:messageBoard`—enables you to read or create a message on the author's message board
- `urn:oracle:webcenter:spaces:profile`—enables you to read a `text/html` document of the author's profile

54.13.1.3 Creating a New Message

Now that you have read the message on your message board, you probably want to reply to Carl on his message board. To do this you should send a `POST` request to the URI for Carl's message board.

To find the correct URI, use the `href` from the `author` link with `resourceType` of `urn:oracle:webcenter:messageBoard`.

A `POST` request creates a subordinate resource of the resource to which you post it. In this case, we are posting to the `messageBoard`, so we should post its subordinate resource: `message` (Example 54–14).

Example 54–14 Creating a Message on Another User's Message Board (POST)

```
POST opaque-messageBoard-URI-for-Carl
Accept: application/json;charset=UTF-8
Content-Type: application/json;charset=UTF-8

{
    "body": "sure; see you guys at 6."
}
```

The response shows that your message was successfully created on Carl's message board (Example 54–15).

Example 54–15 Response to Creating a Message on Another User's Message Board

```
201 Created
Content-Type: application/json;charset=UTF-8

{
  "id": "36b8464f-afda-44b5-90ad-8ecedcb040a3",
  "body": "sure; see you guys at 6.",
  "created": "2009-09-10T12:21:09.785-0700",
  "resourceType": "urn:oracle:webcenter:messageBoard:message",
  "links": [
    {
      "resourceType": "urn:oracle:webcenter:messageBoard:message",
      "capabilities": "urn:oracle:webcenter:read urn:oracle:webcenter:update
urn:oracle:webcenter:delete",
      "rel": "self",
      "href": "opaque-message-URI"
    },
  "author": {
    "id": "mike",
    "displayName": "mike",
    "guid": "649657609D5C11DEBFAA799CBD41D9B8",
    "links": [
      {
        "resourceType": "urn:oracle:webcenter:people:person",
        "capabilities": "urn:oracle:webcenter:read",
        "rel": "self",
        "href": "opaque-person-URI"
      },
      {
        "resourceType": "urn:oracle:webcenter:messageBoard",
        "capabilities": "urn:oracle:webcenter:read urn:oracle:webcenter:create",
        "href": "opaque-messageBoard-URI"
      },
      {
        "type": "text/html",
        "resourceType": "urn:oracle:webcenter:spaces:profile",
        "capabilities": "urn:oracle:webcenter:read",
        "rel": "alternate",
        "href": "opaque:profile:URI"
```

```
      }
    ]
  }
  ]
}
```

54.13.1.4 Updating a Message

A `PUT` request is very similar to a `POST` request, except that it is performed on the resource being edited, instead of on the parent resource.

From the response to your earlier `POST` request, when you created your message on Carl's message board, you can see that you have `read`, `update`, and `delete` capabilities on the message. You can also see that the `href` provides the URI for your message. Something came up at work and you must stay a bit later. Using the URI for your message, you can now send a `PUT` request to update the message and let Carl know that you are going to be late (Example 54–16).

Example 54–16 Updating a Message (PUT)

```
PUT opaque:message:URI
Accept: application/json;charset=UTF-8
Content-Type: application/json;charset=UTF-8

{
    "body": "working late; see you guys at 7."
}
```

The response is nearly identical to that of `POST`, except that the `body` contains your updated message (Example 54–17).

Example 54–17 Updating a Message

```
200 OK
Content-Type: application/json;charset=UTF-8

{
  "id": "36b8464f-afda-44b5-90ad-8ecedcb040a3",
  "body": "working late; see you guys at 7.",

...deleted for brevity...

}
```

54.13.1.5 Deleting a Message

Performing a `DELETE` request on a resource deletes it, if you have the `delete` capability on the resource. The link to your message on Carl's message board supports `delete`.

You decide to delete the message that you left on Carl's message board (Example 54–18).

Example 54–18 Deleting a Message (DELETE)

```
DELETE opaque:message:URI
```

The response is simply a status code of 204 (Example 54–19).

Example 54–19 Response to Deleting a Message

```
204 NO CONTENT
```

> **Note:** `DELETE` is idempotent, meaning that it can be sent multiple times with the same result. Therefore if you try to delete the same object twice, you still receive the same 204 response even though it has previously been deleted.

54.13.2 Displaying Activity Stream Data

To properly display Activity Stream data, you must:

- Retrieve the Activity Stream URI entry point
- Retrieve the Activity Stream data
- Process the Activity Stream data for display



This sample is available on the Oracle WebCenter Portal Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

`http://www.oracle.com/technology/products/webcenter/release11_demos.html`

In Example 54–20:

- The `getResourceURL` method shows you how to retrieve the URI entry point for the Activity Stream service by retrieving the JSON data for the main Resource Index (`/rest/api/resourceIndex`) and locating the URI for the Activity Stream resource in that data.
- The `formatMessage` method processes the Activity Stream data into a displayable format. This involves locating an individual message and replacing any template parameters in the message with the name of the object or user that corresponds to that parameter. The parameters also include links to display the object or user. They may contain links to REST services for those objects or users, if available.

> **Note:** To get the Resource Index and Activity Stream JSON data, make Ajax requests using Javascript (and possibly a client-side scripting library like Dojo) and pass the resulting data into the appropriate methods.

Example 54–20 Displaying Activity Stream Data

```
/* Parse the resourceIndex to find the specified URL and
 * return it.
 *
 * @Param jsonData the JSON data retrieved from calling
 *        the /rest/api/resourceIndex URL.
 * @Param strResourceType the resource type of the URL
 *        you want to retrieve from the resourceIndex data.
 *        E.g., 'urn:oracle:webcenter:activities:stream'
 */
function getResourceURL(jsonData, strResourceType)
{
  // Using the HATEOAS model, we browse the returned links
  // looking for the one with the correct resource type.
```

```
  for (var i = 0; i < data.links.length; i++) {
    if (data.links[i].resourceType == strResourceType) {
      return data.links[i].href;
    }
  }
}


/* Parse the resourceIndex to find the activity stream URL and
 * then load it.
 *
 * @Param jsonData the JSON data retrieved from calling
 *        the /rest/api/resourceIndex URL.
 */
function getActivitiesURL(jsonData)
{
  // Parse the JSON data to get the activities URI entry point.

  var strHref = getResourceURL(jsonData,
    'urn:oracle:webcenter:activities:stream');

  // INSERT CODE HERE: Implement getting the JSON data from the
  // strHref URL with the Accept header set to "application/json",
  // and use the formatMessage(index, jsonData) function to get
  // the displayable activity message for each activity.

}


/* Replace activity message parameters.
 *
 * @Param index the index of the activity to process
 * @Param jsonData the JSON data retrieved from calling
 *        the /rest/api/resourceIndex URL.
 */
function formatMessage(index, jsonData)
{
  var activity = jsonData.items[index];
  var strMessage = activity.message;

  // Look for activity parameters and replace them in the message.

  if (activity.templateParams && activity.templateParams.items) {
    for (var i = 0; i < activity.templateParams.items.length; i++) {
      var param = activity.templateParams.items[i];

      // Each parameter also has a set of links which at least
      // includes an HTML link and possibly a REST API link.

      strMessage = strMessage.replace(param.key, param.displayName);
    }
  }
  if (activity.detail) {
    strMessage = strMessage + "<br><font size='1'>" +
      activity.detail + "</font>";
  }
  return strMessage;
}
```

54.13.3 Using the WebCenter People Connections Service REST APIs with Ext

Ext is a popular JavaScript library for building tables and forms and linking them to REST web services. The following example illustrates how to display a list of connections from the People Connections service REST APIs using an Ext `JsonStore` and `GridPanel`.

For more information about Ext, see:

`http://www.extjs.com/`

> **Note:** You must run Ext on a web server. If that server is not your WebCenter instance, then you will run into cross-site scripting problems. If you cannot run the example on your WebCenter server, and you want to use Ext, then consider a proxy, such as Apache or WebCenter Ensemble.



This sample is available on the Oracle WebCenter Portal Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

`http://www.oracle.com/technology/products/webcenter/release11_demos.html`

This section includes the following subsections:

- Section 54.13.3.1, "Creating the HTML Page"
- Section 54.13.3.2, "Creating the JavaScript File"
- Section 54.13.3.3, "The Result"

54.13.3.1 Creating the HTML Page

Example 54–21 shows the simple HTML page. The HTML page must reference the required Ext files and the JavaScript file, `connections.js`. The body of the page contains a destination `div`. The `GridPanel` is rendered in this `div`.

Example 54–21 The HTML Page

```
<head>
    <link rel="stylesheet" type="text/css" href="../../extjs/resources/css/ext-all.css" />
    <script type="text/javascript" src="../../extjs/adapter/ext/ext-base.js"></script>
    <script type="text/javascript" src="../../extjs/ext-all.js"></script>
    <script type="text/javascript" src="connections.js"></script>
</head>

<body>
    <div id="connections"></div>
</body>
```

54.13.3.2 Creating the JavaScript File

The JavaScript file, `connections.js`, enables the population of the list when the page loads. Example 54–22 shows that all of the JavaScript is a parameter to `Ext.onReady()`.

Example 54–22 connections.js as a Parameter to Ext.onReady

```
Ext.onReady(function() {
        //our code goes here
```

```
});
```

The JavaScript does the following:

1. First, it queries the Resource Index (Example 54–23).

   Notice that the example asks for responses in JSON format.

Example 54–23 Querying the Resource Index

```
//Connection Information
var resourceIndexURL = "http://webcenter_server/rest/api/resourceIndex";
var peopleServiceURN = "urn:oracle:webcenter:people";
var connectionsURN = "urn:oracle:webcenter:people:person:list";
var connectionsRel = "urn:oracle:webcenter:people:person:list:connections";

//create a request for the resourceIndex
Ext.Ajax.request({
        url: resourceIndexURL,
        method: 'GET',
        success: function(response, opts) {
                //locate the people connections entry point
                var index = Ext.decode(response.responseText);
                for (x in index.links) {
                        if (index.links[x].resourceType == peopleServiceURN)
                                getMe(index.links[x].href);
                }
        },
        failure: function(response, opts) {
             alert('Could not communicate with the REST server.');
           },
        headers: {
                'Accept': 'application/json'
        },
});
```

2. Next, the JavaScript sends an authenticated request to the People Connections entry point, and gets the Connections list URL from the response (Example 54–24).

   By default, the REST API uses Basic authentication for secure APIs. In the example, credentials for passing to the Connections list URL are Base64 encoded. Ext does not have a Base64 utility object itself, but one has been developed for it at:

   http://extjs.com/forum/showthread.php?p=167166

Example 54–24 Getting the Connections List URL

```
function getMe(requestURL) {
        //Encode a token to identify yourself with secure parts of the REST API.
        var unencodedToken = "username" + ":" + "password";
        var encodedToken = "Basic " + Ext.util.base64.encode(unencodedToken);

        //call the connections entry point
        Ext.Ajax.request({
                url: requestURL,
                method: 'GET',
                success: function(response, opts) {
                        //locate the link for my connections
                        if (index.links[x].resourceType == connectionsURN) {
                                if (index.links[x].rel == connectionsRel)
                                        getMyConnections(index.links[x].href, encodedToken);
                        }
```

```
                },
                failure: function(response, opts) {
                        alert('Could not communicate with the REST server.');
                },
                headers: {
                        'Accept': 'application/json',
                        'Authorization' : encodedToken
                },
        });
}
```

3. The next step is to send an authenticated request to the Connections list URL (Example 54–25).

Example 54–25 Sending an Authenticated Request to the Connections List URL

```
function getMyConnections(requestURL, encodedToken) {
        Ext.Ajax.request({
                url: requestURL,
                method: 'GET',
                success: function(response, opts) {
                                displayData(response, opts);
                   },
                failure: function(response, opts) {
                     alert('Could not communicate with the REST server.');
                   },
                headers: {
                        'Accept': 'application/json',
                        'Authorization' : encodedToken
                },
        });
}
```

4. Then, the JavaScript adds the response to a `JsonStore` (Example 54–26).

   Previously, this example named `displayData()` as the handler for a successful call to the Connections list URL. Now, it creates the function and adds the `JsonStore` code. In the `JsonStore`, the data model for the data is defined and mapped to results expected from the REST call.

Example 54–26 Adding the Response to a JsonStore

```
function displayData(response, opts) {
        //create a datamodel to display the JSON
        var store = new Ext.data.JsonStore({
                // store configs
                autoDestroy: true,
                storeId: 'forumStore',
                // reader configs
                root: 'items',
                idProperty: 'id',
                fields: ['displayName', {name:'email', mapping:'emails.value'},
                         {name:'phone', mapping:'phoneNumbers[0].value'}]
        });
}
```

5. The JavaScript creates an Ext `GridPanel` to display the `JsonStore` by including the code in Example 54–27 in `displayData()`.

   The grid defines columns and a title. It also includes other display options, such as the size of the table.

Example 54–27 Creating a GridPanel to Display the JsonStore

```
//create the Grid
var grid = new Ext.grid.GridPanel({
    store: store,
    columns: [
        {id: 'conn', header: 'Connection', width: 80, sortable: true, dataIndex: 'displayName'},
        {header: 'Email', width: 120, sortable: true, dataIndex: 'email'},
        {id: 'phone', header: 'Phone', width: 120, sortable: true, dataIndex: 'phone'}
    ],
    stripeRows: true,
    autoExpandColumn: 'conn',
    height: 200,
    width: 400,
    title: 'My Connections',
    viewConfig: {scrollOffset: 2}
});
```

6. Finally, the code renders the `GridPanel` by including the line in Example 54–28 in `displayData()`.

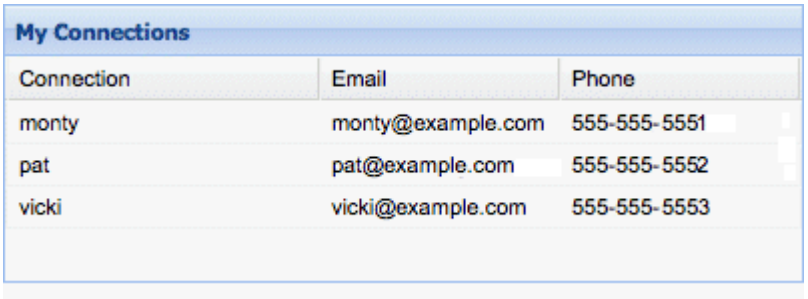
Example 54–28 Rendering the GridPanel

```
grid.render('connections');
```

54.13.3.3 The Result

Figure 54–1 shows the list of connections from the People Connections service REST APIs.

Figure 54–1 The Connections List



54.13.4 Updating User Status

The following example shows how to use REST APIs to update a user's WebCenter profile status.

Note: You must host the sample on a web server (for example, Apache or Oracle HTTP Server) or an application server. To avoid cross site scripting errors, you should proxy URL access to the REST service. On Apache or OHS, the conf commands would look like (change *myspaceshost* and *port* accordingly):

```
ProxyPass /webcenter/ http://myspaceshost:port/webcenter/
ProxyPassReverse /webcenter/ http://myspaceshost:port/webcenter/
ProxyPass /rest/ http://myspaceshost:port/rest/
ProxyPassReverse /rest/ http://myspaceshost:port/rest/
ProxyPreserveHost on
```



This sample is available on the Oracle WebCenter Portal Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

http://www.oracle.com/technology/products/webcenter/release11_demos.html

Updating user status involves making a sequence of asynchronous AJAX calls to get the URL for the status object:

```
urn:oracle:webcenter:resourceindex
   urn:oracle:webcenter:people
      urn:oracle:webcenter:people:person:status
```

The HTML page for updating user status (Example 54–29) includes an input field where users can enter the new status message (`statusMessage`). Clicking the **Update Status** button (or pressing Enter) calls the `updateStatus` method to make the initial call to the Resource Index.

Example 54–29 HTML Body

```
<body>
<div>New status message: <input id="statusMessage" type="text"
   onkeyup="{if (event.keyCode==13) updateStatus();}" maxlength="250"
   size="60" /></div>
<div>
<button id="button1" onclick="updateStatus();">Update Status</button>
</div>
<div id="statusResults"></div>
</body>
```

Figure 54–2 shows the HTML page.

Figure 54–2 New Status Message HTML Page



The code in Example 54–30 retrieves the Resource Index (`resourceIndexURL` variable set to `/rest/api/resourceIndex`). The Resource Index is returned as an AJAX request object (`resourceIndexRequest`).

Example 54–30 Retrieving the Resource Index

```
function updateStatus() {
   // Set the UI to busy state. This is cleared in the success and error callbacks
   setUIBusy("Updating status...");
   //get the Resource Index
   resourceIndexRequest.get(
      resourceIndexURL,
      resourceIndexCallback,
      clearUIBusy);
}
```

The `getResourceURL` method shown in Example 54–31 traverses the links in the data returned by a REST call to find a specified string (URN) that identifies the required resource type.

Example 54–31 Traversing the Links

```
function getResourceURL(data, strResourceType) {
   for (var i = 0; i < data.links.length; i++) {
      if (data.links[i].resourceType == strResourceType) {
         return data.links[i].href;
      }
   }
   return null;
}
```

Example 54–32 uses `getResourceURL` to parse the Resource Index to find the user profile URL. The data is returned as an AJAX request object (`profileRequest`).

Example 54–32 Retrieving the User Profile

```
function resourceIndexCallback(data) {
   //get my user profile
   profileRequest.get(
      getResourceURL(data, 'urn:oracle:webcenter:people'),
      profileCallback,
      clearUIBusy);
}
```

Example 54–33 takes the new status message provided by the user in the HTML page (`statusMessage`) and uses `request.put` to update the status object (retrieved by again calling `getResourceURL`).

Example 54–33 Retrieving the Status Object

```
function profileCallback(data) {
   profile = data;
   // get the URL for the status object
   var url = getResourceURL(data, 'urn:oracle:webcenter:people:person:status');
   // get the new status and escape double quotes
   var newStatus = document.getElementById
      ('statusMessage').value.replace(/\"/g. "\\\"");
   //send a JSON string representing the new status object
   var statusMessage = '{"note": "' + newStatus + '"}';
   statusRequest.put(
      getResourceURL(data, 'urn:oracle:webcenter:people:person:status'),
         statusMessage, renderStatusPutResults, clearUIBusy);
}
```

The `renderStatusPutResults` method, shown in Example 54–34, renders the new status object.

Example 54–34 Rendering the New Status Object

```
function renderStatusPutResults(data) {
   var name = profile.displayName;
   // get the URL for my profile HTML page in WebCenter Portal: Spaces
   var profileURL = getResourceURL(profile,
      'urn:oracle:webcenter:spaces:profile');
   var html = 'Status for <a href="' + profileURL + '" target="_blank">'
      + name + '</a> is: ' + data.note;
   // clear the UI busy state and print the new status message
   clearUIBusy(html);
}
```

Example 54–35 shows the code for disabling and reenabling the UI.

Example 54–35 Disabling the UI

```
function setUIBusy(message) {
   document.getElementById('statusResults').innerHTML = message;
   document.getElementById('button1').disabled = true;
   document.getElementById('statusMessage').disabled = true;
}

function clearUIBusy(message) {
   document.getElementById('statusResults').innerHTML = message;
   document.getElementById('button1').disabled = false;
   document.getElementById('statusMessage').disabled = false;
}
```

Example 54–36 shows the library that assists in AJAX calls. Most of this is not WebCenter specific and works for any XHR requests to a service that returns JSON.

The library includes a reusable XHR object. This object supports HTTP GET, POST, PUT, and DELETE. All functions are asynchronous and take a URL and two callback functions, one for success, and one for failure. Success calls are made with a JavaScript object containing the return data. Failure calls are made with an error string. POST and PUT also take a data argument which must be a JSON string.

Example 54–36 AJAX Library

```
function AjaxRequest() {
   // constructor to create an object oriented AJAX request
   var xhr = null;

   // get XHR object. Should work for IE 6+, Safari, and Mozilla based browsers
   if (window.XMLHttpRequest) {
      xhr = new window.XMLHTTPRequest;
   } else {
      try {
         xhr = new ActiveXObject('MSXML2.XMLHTTP.3.0');
      } catch (ex) {
         xhr = null;
      }
   }
   if (xhr == null) {
      alert("Your browser does not support AJAX");
   }

   this.get = function(url, callback, errorCallback) {
      xhr.open('GET', url, true);
      // the REST APIs return XML by default. Please return JSON
      xhr.setRequestHeader('Accept', 'application/json; charset=utf-8');
      sendRequest(null, callback, errorCallback);
   };

   this.post = function(url, data, callback, errorCallback) {
      xhr.open('POST', url, true);
      // set headers to send and receive JSON
      xhr.setRequestHeader('Accept', 'application/json; charset=utf-8');
      xhr.setRequestHeader('Content-Type', 'application/json; charset=utf-8');
      sendRequest(data, callback, errorCallback);
   };

   this.put = function(url, data, callback, errorCallback) {
      xhr.open('PUT', url, true);
      // set headesr to send and receive JSON
```

```
      xhr.setRequestHeader('Accept', 'application/json; charset=utf-8');
      xhr.setRequestHeader('Content-Type', 'application/json; charset=utf-8');
      sendRequest(data, callback, errorCallback);
   };

   this.deleteResource = function(url, callback, errorCallback) {
      xhr.open('DELETE', url, true);
      sendRequest(null, callback, errorCallback);
   };

   // set the callbacks and send the request with data, if any
   function sendRequest(data, callback, errorCallback) {
      xhr.onreadystatechange = function () {
         processResponse(xhr, callback, errorCallback);
      };
      xhr.send(data);
   }

   function processResponse(xhr, callback, errorCallback) {
      var data = null;
      // can get called here many times. 4 means really done
      if (xhr.readyState == 4) {
         // let's call any HTTP codes in the 200s success
         if (xhr.status >= 200 && xhr.status <300) {
            // convert response text into a JSON object. This is insecure.
            // data should not be blindly evaluated from the server return.
            // consider using json2.js http://www.JSON.org/js.html
            data = eval('(' + xhr.responseText + ')');
            callback(data);
         } else {
            // we got an error. Format an error string
            data = 'Error: ' + xhr.status + ' ' + xhr.statusText + '<br />'
               + xhr.responseText;
            errorCallback(data);
         }
      }
   }
}
```

55

Integrating Other Oracle Applications

This chapter describes how you can integrate other Oracle applications with your WebCenter Portal application.

This chapter contains the following sections:

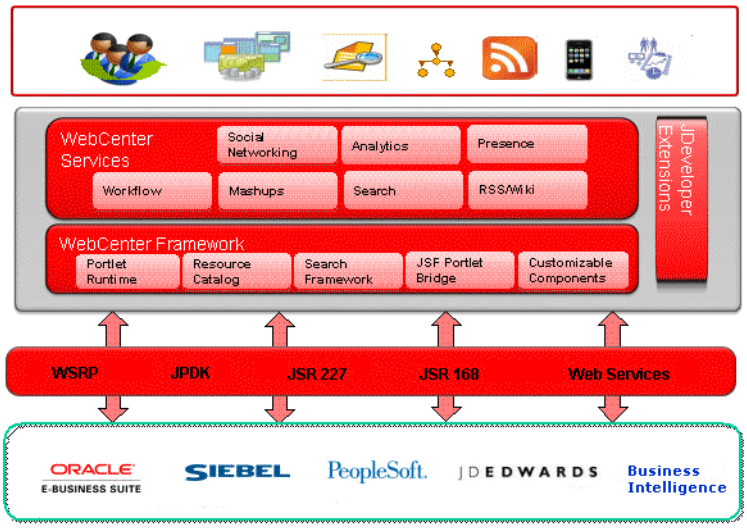
- Section 55.2, "Integrating Siebel Applications"
- Section 55.3, "Integrating E-Business Suite Applications"
- Section 55.4, "Integrating JD Edwards Applications"
- Section 55.5, "Integrating PeopleSoft Applications"
- Section 55.6, "Integrating Oracle Business Intelligence Presentation Services"

55.1 Integrating Oracle and Other Applications in WebCenter Portal

Oracle WebCenter Portal is an integrated suite of technology designed to deliver a unified, context-aware user experience. WebCenter Portal integrates structured and unstructured content, business intelligence, business processes, communication, and collaboration services, and removes the boundaries between enterprise applications. By integrating other applications available within the enterprise with WebCenter Portal, you can create context-centric, composite applications that leverage the capabilities of these applications, extending WebCenter Portal and changing the way people work.

WebCenter Portal uses industry-standard technologies to integrate (primarily as WSRP and JPDK portlets) other application components. Figure 55–1 shows the technologies involved in WebCenter Portal integration with other Oracle applications.

Figure 55–1 WebCenter Portal Integration



Although not all applications support the same integration mechanisms, the integration process is generally quite simple, consisting of exposing the application object to be integrated as a portlet, registering the portlet with WebCenter Portal, adding the portlet to a page, and then running and testing the results.

In Figure 55–1 we show the applications that can be integrated as Siebel, E-Business Suite, JD Edwards, PeopleSoft, and Oracle Business Intelligence. These Oracle applications are fully supported and documented within this chapter. However, you can integrate virtually any application that can expose objects as WSRP or JPDK portlets. The process for integrating them is the same as for the Oracle applications documented here: expose the object as a portlet, register the portlet in WebCenter Portal, and add the portlet to a page. Refer to the documentation for one of the supported Oracle applications for a description of how to consume an exposed portlet in WebCenter Portal.

55.2 Integrating Siebel Applications

This section describes how to integrate a Siebel Web service in a Framework application. It also describes how to integrate Siebel objects using the Siebel Web Engine (SWE). Siebel and WebCenter Portal can work together to include Siebel's CRM capabilities as portlets within your Framework application. You can integrate Siebel applications as Web services, or using the Siebel Web Engine (SWE) as described in the following subsections:

- Section 55.2.1, "Integrating Siebel Applications as Web Services"
- Section 55.2.2, "Integrating Siebel Applications Using the Siebel Web Engine"

55.2.1 Integrating Siebel Applications as Web Services

This section describes how to integrate Siebel applications as Web services in a Framework application.

This section contains the following subsections:

- Section 55.2.1.1, "Preparing the Siebel Application"
- Section 55.2.1.2, "Consuming a Siebel Web Service in a Framework Application"

55.2.1.1 Preparing the Siebel Application

This section describes how to create an inbound Web service, set up operations for the inbound service, and generate a WSDL that you will later use to create a data control to your Framework application.

This section contains the following subsections:

- Section 55.2.1.1.1, "Creating an Inbound Web Service"
- Section 55.2.1.1.2, "Creating Operations for the Inbound Web Service"

55.2.1.1.1 Creating an Inbound Web Service

To create an inbound Web service:

1. Log into the Siebel application.
2. Navigate to the Administration - Web Services page.
3. Click **Inbound Web Services**.

   The Inbound Web Services page shows the out-of-the-box Web services and any other Web services that are currently exposed.
4. Click **Menu** and select `New Record` from the drop-down list.
5. Enter the values for **Namespace**, **Name**, **Status** and **Comment** as appropriate for the Web service you want to set up. For example:

Field Name	Value
Namespace	`http://siebel.com.fmw`
Name	`Siebel Customer Account`
Status	`Active`
Comment	`For Fusion Middleware`

6. Scroll to the Service Ports pane and select **New Record** from the **Menu** drop-down list.
7. Enter `CustAccount` as the **Name** and click **Type**.
8. In the **Inbound Web Service Port Type** pick applet, open the New tab.
9. Select `Business Service` as the **Implementation Type**.
10. From the **Service Name** list, select `Siebel Account`.
11. In the **Inbound Web Service Port Type** pick applet, click **OK** to create the inbound Web service.
12. From the Service Ports dialog's **Transport** drop-down list, select `HTTP`.
13. In the **Address** field, set the URL to your Siebel instance. For example:

```
http://example.com/eai_
enu/start.swe?SWEExtSource=WebService&SWEExtCmd=Execute&UserName=SADMIN&Passwor
d=SADMIN
```

14. From the **Menu** drop-down list, select `Save Record`.

55.2.1.1.2 Creating Operations for the Inbound Web Service

To add operations to the inbound Web service and then create a WSDL file, perform the following steps:

1. Scroll to the Operations section and select `New` from the **Menu** drop-down list.
2. In the **Operation Name** field, enter `AccountInsert`.
3. Click **Method Display Name** to open the Business Service Method dialog.
4. Select `Insert` as the **Method**, and click **OK**.
5. From the **Authentication Type** drop-down list, select an appropriate authentication type:

Authentication Type	Session Type	Description
None	None	A single request is sent with an anonymous user login, and the session is closed after the response is sent out. In order for the anonymous session to be identified by the SWSE Plug-in, UsernameToken and PasswordText must be excluded in the SOAP headers.
Username and password	None	A single request is sent with the username and password used to log in, and the session is closed after the response is sent out.
Username and password	Stateless	The initial request to log in establishes a session that is to remain open and available for subsequent requests. Username/password are used to log in and a session token is returned in a SOAP header included in the outbound response. The session remains open.
Session token (stateless)	Stateless	Request to reconnect to an established session, using the information contained in the session token. If the session has been closed, automatic re-login occurs. The Siebel servers include the session token in the SOAP header of the response. The session remains open.
Session token (stateless)	None	When a SOAP header carries a session token and has the session type set to None, then the Session Manager on the SWSE closes (logs out) of this session, and invalidates the session token. The session token is not used after the session is invalidated.

6. Click **New** to create a new operation.
7. In the **Operation Name** field, enter a name for the new operation (for example, `AccountQueryByExample`).
8. Click **Method Display Name** for the new operation.
9. In the Business Service Method dialog, select the query method (for example, **Query By Example**) and click **OK**.
10. Continue by adding any additional operations you may need as described in steps 6 to 9 above.
11. In the Service Ports pane, select `Save Record` from the **Menu** drop-down list.
12. In the Inbound Web Services pane, select `Save Record` from the **Menu** drop-down list.

13. Select `Clear Cache` from the **Menu** drop-down list.
14. Click the **Generate WSDL**.
15. On the File Download dialog, click **Open**.
16. Select **File --> Save As...**
17. Locate the directory where you want to save the WSDL file, enter a name for the file and click **Save**.

55.2.1.2 Consuming a Siebel Web Service in a Framework Application

This section describes how to consume Siebel applications that have been set up as Web services in a Framework application, including how to set up a WSDL-based data control and, how to use JDeveloper's JSF Navigation Modeler to diagrammatically create your application's pages and the navigation between them.

This section contains the following subsections:

- Section 55.2.1.2.1, "Creating a Data Control Based on a Siebel Web Service"
- Section 55.2.1.2.2, "Creating a Page Flow Diagram"
- Section 55.2.1.2.3, "Adding Pages and Navigation to the Page Flow Diagram"
- Section 55.2.1.2.4, "Creating the Query Page"
- Section 55.2.1.2.5, "Creating the Results Page"
- Section 55.2.1.2.6, "Testing the Application"

55.2.1.2.1 Creating a Data Control Based on a Siebel Web Service

This section describes how to create a WSDL-based data control based on a Web service created from a Siebel application.

To create a WSDL-based data control:

1. Open your Framework application or create a new Framework application in JDeveloper.
2. In the Applications Navigator, right-click the `Model` node and select **New** from the context menu.
3. In the New Gallery, select `All Technologies` from the **Filter By** drop-down list.
4. Select `Business Tier | Web Services` as the category, select the `Web Service Data Control` item, and click **OK**.
5. In the Create Web Service Data Control wizard, click **Next** to leave the Welcome page.
6. In Step 1, enter the Web service name in the **Name** field.
7. Click **Browse** for the **URL** field, locate the directory where you downloaded the WSDL file, select the file and click **Open**.

   In the wizard, note the URL generated in the **Service** field.
8. Click **Next**.

   Step 2 shows all the operations available from the selected Web service.
9. Using the **Add** button, shuttle the operations you want to the Selected pane., and then click **Next**.

When you expand the `Model` node in the Applications Navigator, you should see nodes for the entries that you have created.

10. Click the **Save All** icon to save your changes.

55.2.1.2.2 Creating a Page Flow Diagram -

This section describes how to create the page flow diagram to which you can your add source and query pages.

To create a page flow diagram:

1. In the Applications Navigator, right-click the `ViewController` node and select **New** from the context menu.
2. In the New Gallery, expand the `Web Tier` node and select **JSF**.
3. In the Items pane, select **JSF Page Flow & Configuration (faces-config.xml)** and click **OK**.
4. In the Create JSF Configuration File dialog, click **OK** to accept default values.

   An empty page flow diagram opens with a Component Palette and Data Control Palette to the right of the diagram editor. You use this to create components for the JSF Navigation Model.
5. Click the **Save All** icon to save your changes.

55.2.1.2.3 Adding Pages and Navigation to the Page Flow Diagram

This section describes how to add pages and navigation to the page flow diagram.

To add pages to the page flow diagram:

1. In the JSF Navigation Diagram Component Palette, create a source page by selecting **JSF Page**, clicking the diagram and naming the page appropriately (for example, `QueryByID`). Typically, you will have at least a source and query page.
2. To create the query page, click and drag another JSF Page from the Component Palette and drop it next to the previous one, renaming it appropriately (for example, `ShowResult`).
3. Select **JSF Navigation Case** in the JSF Navigation Diagram Component Palette. Click the icon for the source JSF page, and then click the icon for the destination JSF page for the navigation case.
4. Modify the default label (`success`) by clicking it and entering an appropriate name (for example, `toResult`).
5. Open the Overview tab and click **Navigation Rules**.

   The rule you just created in the diagram should be listed in the table.

   JDeveloper gives you three views of the `faces-config.xml` file: The same information that is presented in the diagram is also accessible through a declarative dialog and directly from the source. If you open the source view (by clicking the Source tab) the `<from-view-id>` tag identifies the source page, and the `<to-view-id>` tag identifies the destination page.
6. From the diagram view, select **JSF Navigation Case** in the Component Palette. Click the icon for the source JSF page, and then click the icon for the destination JSF page for the navigation case.
7. Modify the default label by clicking it and entering an appropriate name (for example, `toQuery`).

8. Click the **Save All** icon to save the diagram.

55.2.1.2.4 Creating the Query Page

This section describes how to create and set up navigation for the query page.

To create the query page:

1. On the Page Flow diagram, double-click the source page icon (`QueryById`) to launch the Create JSF JSP Wizard.
2. Click **Next** to leave the Welcome page.
3. In Step 1 of the wizard, select **JSP Document (*.jspx)** and click **Next**.
4. In Step 2, make sure **Do Not Automatically Expose UI Components in a Managed Bean** is selected and click **Next**.
5. In Step 3, make sure the following libraries are selected:

```
ADF Faces Components 10_1_3_2_0
ADF Faces HTML 10_1_3_2_0
JSF Core 1.0
Cusomizable Components Core 10_1_3_2_0
```

6. Click **Finish** to create the page.

   An empty JSF page opens in the Design tab of the editor.
7. In the Customizable Components Core component palette, click **ShowDetailFrame**.

   The ShowDetailFrame appears in the page.

   Note: Oracle WebCenter Portal provides two customizable components: PanelCustomizable and ShowDetailFrame. These two components make it very easy to organize the content of your Web application, treating different panels as if they are almost stand-alone, portlet-like objects that can be minimized, rearranged, have their own drop-down menus, and so on.
8. With the ShowDetailFrame selected, modify the Text property in the Properties Inspector to Query By ID. Press **[Enter]** to update the page in the Visual Editor.
9. Open the Data Control palette and expand the Siebel Web services node. Drag and drop the `AccountQueryById(String)` node onto the ShowDetailFrame on the page.
10. In the popup menu, select **Create | Parameters | ADF Parameter Form**.
11. In the Edit Form Fields, click **OK**.
12. Select the AccountQueryById button generated in the page and in the Property Inspector pane, change the following properties:

```
Text   Submit
Action toResult
```

13. Select the input field corresponding to the input value, and in the Property Inspector, change the **Label** property to `Account ID:`
14. Click the **Save All** icon to save your work.

55.2.1.2.5 Creating the Results Page

This section describes how to create and add navigation for the results page.

To create the results page:

1. Click the faces-config.xml tab to return to the Page Flow diagram and double-click the **ShowResult** icon to launch the page wizard.
2. Click **Next** to skip the Welcome page.
3. In Step 1 of the wizard, select **JSP Document (*.jspx)** and click **Next**.
4. In Step 2, make sure **Do Not Automatically Expose UI Components in a Managed Bean** is selected and click **Next**.
5. In Step 3, make sure the following libraries are selected:

   ```
   ADF Faces Components 10_1_3_2_0
   ADF Faces HTML 10_1_3_2_0
   JSF Core 1.0
   ```

6. Click **Finish** to create the page.

   An empty JSF page opens in the Design tab of the editor.
7. In the Customizable Components Core component palette, click **ShowDetailFrame**.

   The ShowDetailFrame appears in the page.
8. With the ShowDetailFrame selected, modify the Text property in the Properties Inspector to Query Results. Press **[Enter]** to update the page in the Visual Editor.
9. In the Data Control palette, expand **AccountQueryById | Return | SiebelMessage | ListofAccountInterface**, select the Account node and drop it on the page on the ShowDetailFrame.
10. In the popup menu, select **Create | Forms | ADF Read-only Form**.
11. In the Edit Form Fields, delete all fields except for the following:

    ```
    AccountID
    AccountStatus
    NumberOfEmployees
    CurrencyCode
    Location
    MainPhoneNumber
    Name
    Type
    ```

12. Check the **Include Submit Button** checkbox and click **OK**.
13. In the Action Binding Editor, click **OK**.
14. Click **Submit** and in the Property Inspector change the title to `Back to Query`, and in the **Action** field select `toQuery` from the list.
15. Click the **Save All** icon to save your work.

55.2.1.2.6 Testing the Application

This section describes how to test the application's JSF pages in JDeveloper. Before you can query a Siebel Account using an ID, you need to determine the IDs that are available in your Siebel instance. To perform the following steps you must have access to Siebel Call Center.

To determine the IDS and test your JSF pages:

1. Log into Siebel Call Center, providing the appropriate user ID and password.
2. Navigate to the Accounts List using Site Map.
3. Click **Accounts**, then **Accounts List**.
4. From the returned list, highlight the account you want to query.
5. From the Help menu, select **About Record**.
6. Find the ID value in the **Row #** field and make a note of it.
7. Return to the page flow diagram and right-click the QueryById page icon and select **Run** from the context menu.

   The page is loaded in your default browser.
8. In the ID field, enter the value you previously noted in the the **Row #** field and click **Submit**.

   The Details page displays the detail information for the corresponding account.

55.2.2 Integrating Siebel Applications Using the Siebel Web Engine

As well as integrating Siebel applications using Web Services, you can also integrate Siebel applications in a Framework application using the Siebel Web Engine (SWE).

Siebel Web Engine (SWE) is responsible for rendering the Siebel User Interface. Siebel Web Templates provide this HTML layout information (markup information) to the Siebel Web Engine when rendering Siebel objects in the repository definition of the application. The markup that SWE returns can also be XML for rendering within XML-aware applications or WML (wireless markup language) for rendering on wireless devices. This lets you request the SWE to return a Siebel View as XML, parse the data elements, and display the result in a Framework application.

For instructions on how to use SWE within a Framework application, follow the tutorial *Using the Siebel Web Engine (SWE) to View Data in a WebCenter Application*.

To complete the tutorial, you will need the following:

- Have access to or have installed Oracle JDeveloper Studio Edition
- Have access to or have installed Oracle's Siebel eBusiness Applications.
- Have access to a supported version of Windows Internet Explorer (IE) (see the *System Requirements and Supported Platforms Guide* for Siebel that is available on OTN).

Also refer to the Siebel Bookshelf Guides: *Siebel Portal Framework Guide* (specifically, the section on delivering content to external Web applications), and *Configuring Siebel Business Applications Guide* for details on SWE and Web Templates.

55.3 Integrating E-Business Suite Applications

This section describes how to integrate E-Business Suite applications in WebCenter Portal applications.

This section contains the following subsections:

- Section 55.3.1, "Introduction to Integrating EBS Applications"
- Section 55.3.2, "Integrating EBS Applications in a Framework Application"

- Section 55.3.3, "Integrating EBS Applications in Spaces"

55.3.1 Introduction to Integrating EBS Applications

This section describes the integration points and requirements integrating Oracle E-Business Suite portlets in WebCenter Portal applications.

This section includes the following subsections:

- Section 55.3.1.1, "Understanding EBS Integration"
- Section 55.3.1.2, "Requirements for Integrating EBS Applications"

55.3.1.1 Understanding EBS Integration

Out-of-the-box, Oracle E-Business Suite OA Framework-based portlets, such as Applications Navigator, Favorites, and Worklist are WSRP and JSR 168-compliant. That means that you can access these Oracle E-Business Suite portlets from WSRP-compliant portal servers such as WebCenter Portal: Framework applications or WebCenter Portal: Spaces, by simply adding the portlet onto an application page. Follow the instructions in Section 55.3.2.3, "Creating a JSF Page to Consume the Remote Producer" to add them to a Framework application page, or Section 55.3.3.3, "Adding the EBS Portlet to a Spaces Page" to add them to a Spaces page.

You can also create new E-Business Suite portlets that are WSRP and JSR 168-compliant that can similarly be added to a Framework application. Creating and consuming WSRP and JSR 168 compliant portals in WebCenter Portal is described in Section 55.3.2, "Integrating EBS Applications in a Framework Application" and Section 55.3.3, "Integrating EBS Applications in Spaces."

55.3.1.2 Requirements for Integrating EBS Applications

The following requirements apply for integrating Oracle E-Business Suite portals into WebCenter Portal applications:

- Regions to be exposed as portlets must be created using Oracle E-Business Suite OA Framework Release 12 as previous versions are not WSRP/JSR 168-compliant.
- Oracle E-Business Suite should be configured to use Oracle Internet Directory (OID) and one of following single sign-on solutions:
  - Oracle Single Sign-On (OSSO)
  - Oracle Access Manager (OAM)

  If you are using OSSO, follow the steps in My Oracle Support document 376811.1 to integrate E-Business Suite Release 12 with OID and OSSO.

  If you are using OAM, follow the steps in My Oracle Support document 975182.1 to integrate E-Business Suite Release 12 with OAM.
- You must have granted the WebCenter Portal application access to the E-Business Suite Portlet Producer and added and configured the appropriate users.

> **Note:** To complete some steps, you may need system administrator permissions.

55.3.2 Integrating EBS Applications in a Framework Application

This section includes the following subsections:

- Section 55.3.2.1, "Preparing the EBS Application for Remote Access"
- Section 55.3.2.2, "Registering the EBS WSRP Producer in the Framework Application"
- Section 55.3.2.3, "Creating a JSF Page to Consume the Remote Producer"
- Section 55.3.2.4, "Testing the Framework Application"

55.3.2.1 Preparing the EBS Application for Remote Access

Prepare the standalone regions to be portletized as described in the section on "Basic Implementation" under "Portlets" in Chapter 4: "Implementing Specific UI Features" in the *Oracle Application Framework Developer's Guide*, available from My Oracle Support at:

```
https://support.us.oracle.com/oip/faces/secure/km/DocumentDisplay.jspx?id=1107973.1
```

The Oracle E-Business Suite OA Framework lets you expose any standalone regions (except pageLayout regions) as portlets.

You can also use an Oracle E-Business Suite tool called Portlet Generator to convert existing standalone Oracle Application Framework regions into portlets. For more information, see the section on "Portlet Generator" in Chapter 4: "Developer Tools" in the *Oracle Application Framework Developer's Guide*.

After creating the portlets, grant access to your new UI Function by adding it to a menu and associating the menu with a responsibility. For more information, see the section on "Page Security" under "Portlets" in Chapter 4: "Implementing Specific UI Features" in the *Oracle Application Framework Developer's Guide*.

Before adding the portlets in WebCenter Portal, be sure to bounce the Apache listener as the menu and function definitions are cached. Note that you'll need system administrator permissions to do this.

55.3.2.2 Registering the EBS WSRP Producer in the Framework Application

Follow the instructions below to create or open an existing Framework application, and register the EBS WSRP producer.

1. Create a new Framework application or open an existing one in which you would like to consume the remote (WSRP) EBS portlet.
2. Under Application Resources, right-click **Connection** and select **WSRP Producer**.

   Page 1 (Specify Producer Name) of the Register WSRP Portlet Producer wizard displays.
3. Enter a name for the WSRP producer and click **Next**.

   The Specify Connection Details page displays.
4. Enter the WSDL for the EBS producer in the **WSDL URL** field as shown below:

   ```
   WSDL URL: http://[Release_12_host]:[port]/OA_HTML/portlets/WSRPBaseService?WSDL
   ```

   For example:

   ```
   http://myEBSServer.example.com:8001/OA_HTML/portlets/WSRPBaseService?WSDL
   ```
5. Click **Next** and then **Finish**.

   The WSRP producer should now appear under Connections.

6. Continue by creating a JSF page to consume the WSRP producer as described in Section 55.3.2.3, "Creating a JSF Page to Consume the Remote Producer."

55.3.2.3 Creating a JSF Page to Consume the Remote Producer

Use the following procedure to add or use an existing JSF page to consume the EBS remote producer in your Framework application.

To create a JSF page:

1. In the application's Navigation panel, right-click on the application name and select **New**.

   The New Gallery dialog displays.

2. In the Categories field's tree structure, locate and expand the Web Tier.

   A list with the descriptions of the available options display in the Items field.

3. From this list, select **JSF Page** and click **OK**.

   The Create JSF Page dialog displays.

4. Create the JSF page, making sure that the `Create an XML Document (*.jspx)` field is selected. For more information about setting up the JSF page, click the Help button to access the online help.

5. Click **OK**.

   The newly created JSF page displays.

6. Under connections, expand the newly created WSRP producer and drag the portlet for the EBS page onto the JSF page.

7. In the Portlet Property Inspector, set the **RenderPortletInIFrame** property to `True`.

8. Add any other components that you need onto the page, and save the application.

9. Continue by running the page and testing that modifications made in the Framework application appear in the EBS application as described in Section 55.3.2.4, "Testing the Framework Application."

55.3.2.4 Testing the Framework Application

Use this procedure to test the Framework application by modifying content on the JSF page in the Framework application and checking that the modification shows up in the EBS application.

1. Run the `.jspx` page that you created.

2. In the running page, modify some information that you can verify the changes for in the EBS application.

3. Save your changes and confirm that the changes also appear in the EBS application.

55.3.3 Integrating EBS Applications in Spaces

This section contains the following subsections:

- Section 55.3.3.1, "Preparing the EBS Portlet for Remote Access"
- Section 55.3.3.2, "Registering the EBS WSRP Producer in Spaces"
- Section 55.3.3.3, "Adding the EBS Portlet to a Spaces Page"

- Section 55.3.3.4, "Testing the Spaces Portlet Connection"

55.3.3.1 Preparing the EBS Portlet for Remote Access

Prepare the standalone regions to be portletized as described the section on "Basic Implementation" under "Portlets" in Chapter 4: "Implementing Specific UI Features" in the *Oracle Application Framework Developer's Guide*, available from My Oracle Support at:

`https://support.us.oracle.com/oip/faces/secure/km/DocumentDisplay.jspx?id=1107973.1`

The Oracle E-Business Suite OA Framework lets you expose any standalone regions (except pageLayout regions) as portlets.

You can also use an Oracle E-Business Suite tool called Portlet Generator to convert existing standalone Oracle Application Framework regions into portlets. For more information, see the section on "Portlet Generator" in Chapter 4: "Developer Tools" in the *Oracle Application Framework Developer's Guide*.

After creating the portlets, grant access to your new UI Function by adding it to a menu and associating the menu with a responsibility. For more information, see the section on "Page Security" under "Portlets" in Chapter 4: "Implementing Specific UI Features" in the *Oracle Application Framework Developer's Guide*.

Before adding the portlets in Spaces, be sure to bounce the Apache listener as the menu and function definitions are cached.

55.3.3.2 Registering the EBS WSRP Producer in Spaces

You can register the EBS WSRP producer directly in Spaces, as described in the section on "Registering Portlet Producers" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. You can also register the EBS WSRP producer using Fusion Middleware Control as described in the steps below.

To register the EBS WSRP producer using Fusion Middleware Control:

1. Prepare the EBS page that you want to consume in Spaces for remote access as described in Section 55.3.3.1, "Preparing the EBS Portlet for Remote Access."
2. Log into Fusion Middleware Control for the Spaces domain.
3. Expand `Spaces` in the Navigation bar and from the WebCenter Portal menu, and select **Register Producer**.

   The Add Portlet Producer page displays.
4. Enter a **Connection Name**, set the **Producer Type** to `WSRP Producer`, and paste the WSDL endpoint URL that you copied in step 1 into the **URL End Point** field.
5. Click **OK** and verify that the producer connection was created successfully.
6. Continue by adding the portlet to a Spaces page as described in Section 55.3.3.3, "Adding the EBS Portlet to a Spaces Page."

55.3.3.3 Adding the EBS Portlet to a Spaces Page

Use the following procedure to consume the EBS remote producer in Spaces.

1. Log into Spaces.
2. Go to the page, or create a new page, where you want to add the EBS portal.
3. Click **Add Content** to open Composer.

4. From the Resource Catalog, select **Portlets**.

   Note that if you've created a custom catalog, **Portlets** may not appear, in which case you will need to add it to the Resource Catalog. For information about managing Resource Catalogs, see the chapter "Working with Resource Catalogs" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

5. Click the portlet you added in Fusion Middleware Control.
6. Click **Add** for the EBS portlet you want to add to your Spaces page.
7. Click **Close** to close Composer.
8. On the Spaces page, click the **Edit** (pencil) icon.
9. Open the Display Options tab and set the **Render Portlet in IFrame** option to `True` and click **OK**.
10. Continue by checking the portlet connection as described in Section 55.3.3.4, "Testing the Spaces Portlet Connection."

55.3.3.4 Testing the Spaces Portlet Connection

Use this procedure to test the Spaces portlet connection by modifying content and checking that the modification shows up in the EBS application.

1. On the Spaces page that you added the EBS portlet to, modify some information that you can verify the changes for in the EBS application.
2. Save your changes and confirm that the changes also appear in the EBS application.

55.4 Integrating JD Edwards Applications

This section describes how to integrate JD Edwards applications into WebCenter Portal applications.

This section contains the following subsections:

- Section 55.4.1, "Integrating JD Edwards Applications in a Framework Application"
- Section 55.4.2, "Integrating JD Edwards Applications in Spaces"

55.4.1 Integrating JD Edwards Applications in a Framework Application

This section includes the following subsections:

- Section 55.4.1.1, "Preparing the JD Edwards Application for Remote Access"
- Section 55.4.1.2, "Registering the JD Edwards WSRP Producer in the Framework Application"
- Section 55.4.1.3, "Creating the JSF Page to Consume the Remote Producer"
- Section 55.4.1.4, "Testing the Framework Application"

55.4.1.1 Preparing the JD Edwards Application for Remote Access

Before you can add JD Edwards standalone regions to Framework applications, you must first prepare them to be portletized within JD Edwards by making them available externally as portlets and locating the pre-configured WSDL in the

`webclient.war/wsdl` directory. The WSDL URL is needed so that you can register the JD Edwards WSRP producer and consume it from a Framework application page.

55.4.1.2 Registering the JD Edwards WSRP Producer in the Framework Application

Follow the instructions below to create or open an existing Framework application, and register the JD Edwards WSRP producer.

1. Create a new Framework application or open an existing one in which you would like to consume the remote (WSRP) JD Edwards portlet.
2. Under Application Resources, right-click Connection and select **WSRP Producer**.

   Page 1 (Specify Producer Name) of the Register WSRP Portlet Producer wizard displays.
3. Enter a name for the WSRP producer and click **Next**.

   The Specify Connection Details page displays.
4. Paste the WSDL URL for the JD Edwards producer into the **WSDL URL** field.
5. Click **Next** and then **Finish**.

   The WSRP producer should now appear under Connections.
6. Continue by creating a JSF page to consume the WSRP producer as described in Section 55.4.1.3, "Creating the JSF Page to Consume the Remote Producer."

55.4.1.3 Creating the JSF Page to Consume the Remote Producer

Use the following procedure to add or use an existing JSF page to consume the JD Edwards remote producer in your Framework application.

To create a JSF page:

1. In the application's Navigation panel, right-click on the application name and select **New**.

   The New Gallery dialog displays.
2. In the Categories field's tree structure, locate and expand the Web Tier.

   A list with the descriptions of the available options display in the Items field.
3. From this list, select **JSF Page** and click **OK**.

   The Create JSF Page dialog displays.
4. Create the JSF page, making sure that the `Create an XML Document (*.jspx)` field is selected. For more information about setting up the JSF page, click the Help button to access the online help.
5. Click **OK**.

   The newly created JSF page displays.
6. Under connections, expand the newly created WSRP producer and drag the portlet for the JD Edwards page onto the JSF page.
7. In the Portlet Property Inspector, set the **RenderPortletInIFrame** property to `True`.
8. Add any other components that you need onto the page, and save the application.
9. Continue by running the page and testing that modifications made in the Framework application appear in the JD Edwards application as described in Section 55.4.1.4, "Testing the Framework Application."

55.4.1.4 Testing the Framework Application

Use this procedure to test the Framework application by modifying content on the JSF page in the Framework application and checking that the modification shows up in the JD Edwards application.

1. Run the `.jspx` page that you created.
2. In the running page, modify some information that you can verify the changes for in the JD Edwards application.
3. Save your changes and confirm that the changes also appear in the JD Edwards application.

55.4.2 Integrating JD Edwards Applications in Spaces

This section contains the following subsections:

- Section 55.4.2.1, "Registering the Producer"
- Section 55.4.2.2, "Adding the JD Edwards Portlet to a Spaces Page"
- Section 55.4.2.3, "Testing the Spaces Portlet Connection"

55.4.2.1 Registering the Producer

You can register the JD Edwards WSRP producer directly in Spaces, as described in the section on "Registering Portlet Producers" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. You can also register the JD Edwards WSRP producer using Fusion Middleware Control as described in the steps below.

To register the JD Edwards WSRP producer using Fusion Middleware Control:

1. Prepare the JD Edwards page that you want to consume in Spaces for remote access as described in Section 55.4.1.1, "Preparing the JD Edwards Application for Remote Access."
2. Log into Fusion Middleware Control for the Spaces domain.
3. Expand `Spaces` in the Navigation bar and from the WebCenter Portal menu, and select **Register Producer**.

   The Add Portlet Producer page displays.
4. Enter a **Connection Name**, set the **Producer Type** to `WSRP Producer`, and paste the WSDL endpoint URL that you copied in step 1 into the **URL End Point** field.
5. Click **OK** and verify that the producer connection was created successfully.
6. Continue by adding the portlet to a Spaces page as described in Section 55.4.2.2, "Adding the JD Edwards Portlet to a Spaces Page."

55.4.2.2 Adding the JD Edwards Portlet to a Spaces Page

Use the following procedure to consume the JD Edwards remote producer in Spaces.

1. Log into Spaces.
2. Go to the page, or create a new page, where you want to add the JD Edwards portal.
3. Click **Add Content** to open Composer.
4. From the Resource Catalog, select **Portlets**.

Note that if you've created a custom catalog, **Portlets** may not appear, in which case you will need to add it to the Resource Catalog. For information about managing Resource Catalogs, see the chapter "Working with Resource Catalogs" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

5. Click the portlet you added in Fusion Middleware Control.
6. Click **Add** for the JD Edwards portlet you want to add to your Spaces page.
7. Click **Close** to close Composer.
8. On the Spaces page, click the **Edit** (pencil) icon.
9. Open the Display Options tab and set the **Render Portlet in IFrame** option to `True` and click **OK**.
10. Continue by checking the portlet connection as described in Section 55.4.2.3, "Testing the Spaces Portlet Connection."

55.4.2.3 Testing the Spaces Portlet Connection

Use this procedure to test the Spaces portlet connection by modifying content and checking that the modification shows up in the JD Edwards application.

1. On the Spaces page that you added the JD Edwards portlet to, modify some information that you can verify the changes for in the JD Edwards application.
2. Save your changes and confirm that the changes also appear in the JD Edwards application.

55.5 Integrating PeopleSoft Applications

This section describes how to integrate PeopleSoft applications in a WebCenter Portal application.

This section contains the following subsections:

- Section 55.5.1, "Introduction to Integrating PeopleSoft Applications"
- Section 55.5.2, "Integrating PeopleSoft Applications in a Framework Application"
- Section 55.5.3, "Integrating PeopleSoft Applications in Spaces"

55.5.1 Introduction to Integrating PeopleSoft Applications

This section describes the benefits and methods involved in integrating PeopleSoft applications into WebCenter Portal applications.

This section includes the following subsections:

- Section 55.5.1.1, "Understanding PeopleSoft Integration"
- Section 55.5.1.2, "Requirements for Integrating PeopleSoft Applications"

55.5.1.1 Understanding PeopleSoft Integration

PeopleTools 8.48 and later lets you expose PeopleSoft applications as WSRP portlets in remote applications such as WebCenter Portal. This allows people who only need access to a small portion of PeopleSoft's functionality to access it through a WebCenter Portal application without needing to open or learn the entire PeopleSoft application.

55.5.1.2 Requirements for Integrating PeopleSoft Applications

This section the prerequisites for integrating PeopleSoft objects in Framework applications, and Spaces.

- PeopleTools 8.48 or later
- When using WS-Security for automatic sign on to PeopleSoft, in order for the SAML assertion to be valid, the date/time on the PeopleSoft and WebCenter Portal servers must be synchronized. If this is problematic, then the PeopleSoft web server's time may be set to be slightly ahead of WebCenter Portal server.
- For PeopleTools 8.5.1 and earlier, you may need to create and configure a custom OWSM policy in order to fully support WS-Security. For more information, see Section 55.5.2.4, "Setting up WS-Security for PeopleTools."

55.5.2 Integrating PeopleSoft Applications in a Framework Application

This section describes how to create or modify a Framework application that integrates with PeopleSoft Applications.

This section includes the following subsections:

- Section 55.5.2.1, "Preparing the PeopleSoft Application for Remote Access"
- Section 55.5.2.2, "Registering the PeopleSoft WSRP Producer in the Framework Application"
- Section 55.5.2.3, "Creating the JSF Page to Consume the Remote (WSRP) Producer"
- Section 55.5.2.4, "Setting up WS-Security for PeopleTools"
- Section 55.5.2.5, "Testing the Framework Application"

55.5.2.1 Preparing the PeopleSoft Application for Remote Access

This section describes how to prepare the PeopleSoft application so that it can be consumed by Spaces or a Framework application.

To prepare the PeopleSoft application:

1. Log into PeopleSoft as an administrator.
2. Select PeopleTools from the main menu.
3. From the People Tools main menu, expand **Employee Self-Service**, and then **Portal**.
4. Select **Structure and Content**.
5. Navigate to the folder containing the page that you want to consume in WebCenter Portal and click **Edit** to open it.

   The Content Ref Administration page displays.
6. On the General tab, select the **WSRP Producible** checkbox.
7. Save the page.
8. From the main menu, select **WSRP Production**.

   The Producer Offered Portlets page displays.
9. Verify that the page is exposed, and then expand Web Service Endpoint URL and copy the URL (the WSDL).

10. Open a new tab in your browser, and paste the copied URL into the Navigation Bar to access the WSDL page.
11. Copy the URL to the clipboard.
12. Continue by creating or modifying a Framework application that can access the PeopleSoft remote (WSRP) producer as described in Section 55.5.2.2, "Registering the PeopleSoft WSRP Producer in the Framework Application."

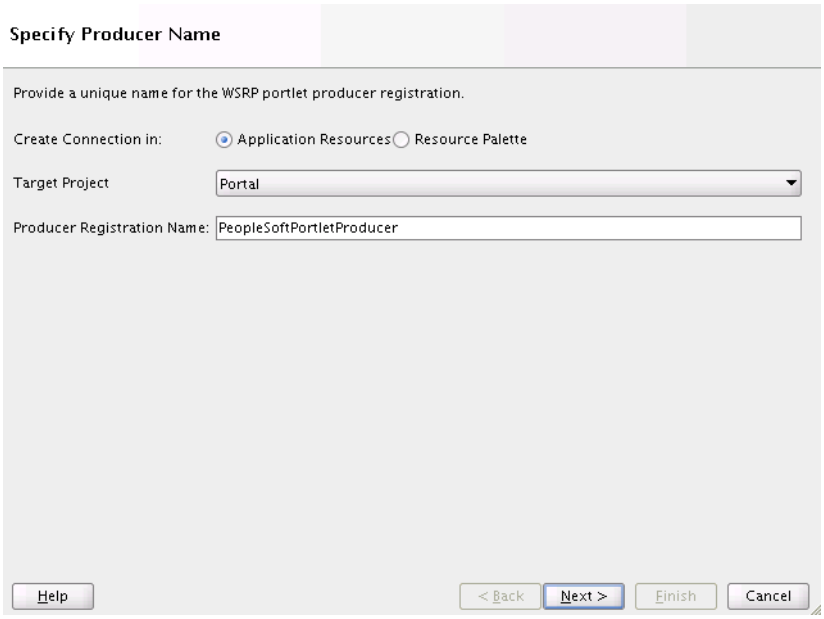
55.5.2.2 Registering the PeopleSoft WSRP Producer in the Framework Application

Follow the instructions below to create or open an existing Framework application, and register the PeopleSoft WSRP producer.

1. Create a new Framework application or open an existing one in which you would like to consume the remote (WSRP) PeopleSoft portlet.
2. Under Application Resources, right-click Connection and select **WSRP Producer**.

   Page 1 (Specify Producer Name) of the Register WSRP Portlet Producer wizard displays (see Figure 55–2).

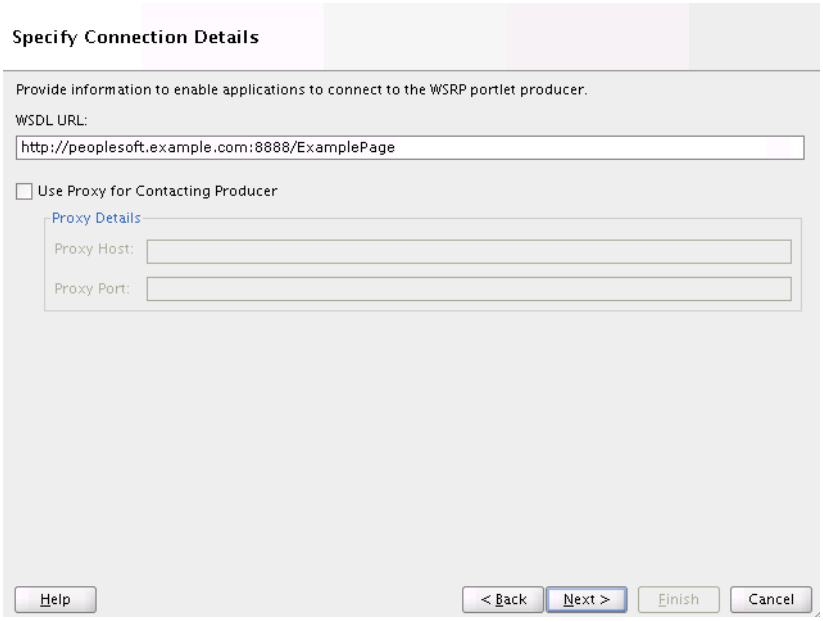
Figure 55–2 Specify Producer Name



3. Enter a name for the WSRP producer and click **Next**.

   The Specify Connection Details page displays (see Figure 55–3).

Figure 55–3 Specify Connection Details



4. Paste the WSDL (the URL you copied earlier) into the **WSDL URL** field.
5. Click **Next** and then **Finish**.

   The WSRP producer should now appear under Connections.
6. Continue by creating a JSF page to consume the WSRP producer as described in Section 55.5.2.3, "Creating the JSF Page to Consume the Remote (WSRP) Producer."

Note: Compression must be turned off in WLS to expose WSRP Portlets. Compression is enabled by default in the Producer Web Server running with Weblogic 10.3.2 and must be disabled in the Server Manager setting for compression available for each instance.

55.5.2.3 Creating the JSF Page to Consume the Remote (WSRP) Producer

Use the following procedure to add a JSF page to consume the PeopleSoft remote producer.

To create a JSF page:

1. In the application's Navigation panel, right-click on the application name and select **New**.

   The New Gallery dialog displays.
2. In the Categories field's tree structure, locate and expand the Web Tier.

   A list with the descriptions of the available options display in the Items field.
3. From this list, select **JSF Page** and click **OK**.

   The Create JSF Page dialog displays.
4. Create the JSF page, making sure that the `Create an XML Document (*.jspx)` field is selected. For more information about setting up the JSF page, click the Help button to access the online help.
5. Click **OK**.

The newly created JSF page displays.

6. Under connections, expand the newly created WSRP producer and drag the portlet for the PeopleSoft page onto the JSF page.
7. In the Portlet Property Inspector, set the **RenderPortletInIFrame** property to `True`.
8. Add any other components that you need onto the page, and save the application.
9. Continue by running the page and testing that modifications made in the Framework application appear in the PeopleSoft application as described in Section 55.5.2.5, "Testing the Framework Application."

55.5.2.4 Setting up WS-Security for PeopleTools

PeopleTools release 8.5.1 (and earlier) does not support outgoing WS-Security headers in its messages. The out-of-the-box WebCenter Portal/OWSM policies require that both outgoing and incoming messages be secured. To bridge this gap you may need to create custom OWSM policies. This section describes the different integration scenarios that require custom WS-Security policies, and the steps required on WebCenter Portal side to configure them.

This section includes the following subsections:

- Section 55.5.2.4.1, "Configuring WS-Security for WSS10 SAML Token with Message Integrity"
- Section 55.5.2.4.2, "Configuring WS-Security for WSS10 SAML Token with Message Protection"
- Section 55.5.2.4.3, "Configuring WS-Security for WSS10 Username Token with Password"

55.5.2.4.1 Configuring WS-Security for WSS10 SAML Token with Message Integrity

This section describes how to configure WS-Security for the WSS10 SAML Token with Message Integrity (`oracle/wss10_saml_token_with_message_integrity_client_policy`) policy.

To configure WS-Security:

1. Configure the WebCenter Portal/OWSM keystore as shown in the chapter "Configuring WS-Security" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
2. Generate a certificate containing the public key of the WebCenter Portal domain and send it to the PeopleTools administrator so it can be imported in the PeopleTools configuration.
3. When you register the producer, choose `wss10_saml_token_with_message_integrity_client_policy`.
4. Continue by adding the WSRP portlet to a WebCenter Portal page.

55.5.2.4.2 Configuring WS-Security for WSS10 SAML Token with Message Protection

The default WSS10 SAML Token with Message Protection (`oracle/wss10_saml_token_with_message_protection_client_policy`) policy that ships with OWSM requires that response also be signed and encrypted. However, PeopleTools release 8.5.1 and earlier cannot send WS-Security headers in response (only the initial `cookie/get portlet handle` call contains security headers; subsequent calls do not) and we therefore need to create and attach a custom policy based on the

oracle/wss10_saml_token_with_message_protection_client_policy policy.

To create a custom policy:

1. Log into Fusion Middleware Control and navigate to the WebCenter Portal domain (wc_domain by default).
2. From the WebLogic Domain menu, select **Web Services > Policies**.
3. Select the wss10_saml_token_with_message_protection_client_policy and click **Create Like**.
4. Give the policy a new name (for example, oracle/wss10_saml_token_with_message_protection_plaintext_response_client_policy).
5. Open the Response tab, uncheck the **Include Entire Body** checkboxes under Message Signing Setting and Message Encrypt Setting , and save the policy.
6. Use WLST to register the producer using the newly created policy as shown in the following example:

```
registerWSRPProducer('webcenter', 'wc-pt851-saml_msg-protection',
'http://example.com/pspc/pswsdl/ps/EMPLOYEE', timeout=100,
tokenType='oracle/wss10_saml_token_with_message_protection_plaintext_response_
client_policy',
enforcePolicyURI='false', issuer='www.oracle.com',
sigKeyAlias='webcenter',sigKeyPswd='welcome1', encKeyAlias='webcenter',
encKeyPswd='welcome1', recptAlias='peopleTools_public')
```

Note: You must use WLST to register the producer. Fusion Middleware Control can only accept fixed policy names and therefore you must register the producer with this policy using WLST by passing in enforcePolicyURI='false'.

55.5.2.4.3 Configuring WS-Security for WSS10 Username Token with Password

The default WSS10 Username Token with Password (oracle/wss10_username_token_with_message_protection_client_policy) policy that ships with OWSM requires that response also be signed and encrypted. However, PeopleTools release 8.5.1 and earlier cannot send WS-Security headers in response (only the initial cookie/get portlet handle call contains security headers; subsequent calls do not) and we therefore need to create and attach a custom policy based on the oracle/wss10_username_token_with_message_protection_client_policy policy.

To create a custom policy:

1. Log into Fusion Middleware Control and navigate to the WebCenter Portal domain (wc_domain by default).
2. From the WebLogic Domain menu, select **Web Services > Policies**.
3. Select the wss10_username_token_with_message_protection_client_policy and click **Create Like**.
4. Give the policy a new name (for example, oracle/wss10_username_token_with_message_protection_plaintext_response_client_policy).
5. Open the Response tab, uncheck the **Include Entire Body** checkboxes under Message Signing Setting and Message Encrypt Setting , and save the policy.
6. Use WLST to register the producer using the newly created policy as shown in the following example:

```
registerWSRPProducer('webcenter', 'wc-pt851-saml_msg-protection',
'http://example.com/pspc/pswsdl/ps/EMPLOYEE', timeout=100,
tokenType='oracle/wss10_username_token_with_message_protection_plaintext_
response_client_policy',
enforcePolicyURI='false', issuer='www.oracle.com',
sigKeyAlias='webcenter',sigKeyPswd='welcome1', encKeyAlias='webcenter',
encKeyPswd='welcome1', recptAlias='peopleTools_public')
```

Note: You must use WLST to register the producer. Fusion Middleware Control can only accept fixed policy names and therefore you must register the producer with this policy using WLST by passing in `enforcePolicyURI='false'`.

55.5.2.5 Testing the Framework Application

Use this procedure to test the Framework application by modifying content on the JSF page in the Framework application and checking that the modification shows up in the PeopleSoft application.

1. Run the `.jspx` page that you created.
2. In the running page, modify some information that you can verify the changes for in the PeopleSoft application.
3. Save your changes and confirm that the changes also appear in the PeopleSoft application.

55.5.3 Integrating PeopleSoft Applications in Spaces

This section describes how to integrate a PeopleSoft application in Spaces.

This section contains the following subsections:

- Section 55.5.3.1, "Registering the Producer"
- Section 55.5.3.2, "Adding the PeopleSoft Portlet to a Spaces Page"
- Section 55.5.3.3, "Testing the Spaces Portlet Connection"

55.5.3.1 Registering the Producer

You can register the PeopleSoft WSRP producer directly in Spaces, as described in the section on "Registering Portlet Producers" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*. You can also register the PeopleSoft WSRP producer using Fusion Middleware Control as described in the steps below.

To register the PeopleSoft WSRP producer using Fusion Middleware Control:

1. Prepare the PeopleSoft page that you want to consume in Spaces for remote access as described in Section 55.5.2.1, "Preparing the PeopleSoft Application for Remote Access."
2. Log into Fusion Middleware Control for the Spaces domain.
3. Expand `Spaces` in the Navigation bar and from the WebCenter Portal menu, select **Register Producer**.

   The Add Portlet Producer page displays.
4. Enter a **Connection Name**, set the **Producer Type** to `WSRP Producer`, and paste the WSDL endpoint URL that you copied in step 1 into the **URL End Point** field.
5. Click **OK** and verify that the producer connection was created successfully.

6. Continue by adding the portlet to a Spaces page as described in Section 55.5.3.2, "Adding the PeopleSoft Portlet to a Spaces Page."

55.5.3.2 Adding the PeopleSoft Portlet to a Spaces Page

Follow the steps below to add the PeopleSoft portlet to a Spaces page.

1. Log into Spaces.
2. Go to the page, or create a new page, where you want to add the PeopleSoft portal.
3. Click **Add Content** to open Composer.
4. From the Resource Catalog, select **Portlets**.

   Note that if you've created a custom catalog, **Portlets** may not appear, in which case you will need to add it to the Resource Catalog. For information about managing Resource Catalogs, see the chapter "Working with Resource Catalogs" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
5. Click the portlet you added in Fusion Middleware Control.
6. Click **Add** for the PeopleSoft page you want to add to your Spaces page.
7. Click **Close** to close Composer.
8. On the Spaces page, click the **Edit** (pencil) icon.
9. Open the Display Options tab and set the **Render Portlet in IFrame** option to `True` and click **OK**.
10. Continue by checking the portlet connection as described in Section 55.5.3.3, "Testing the Spaces Portlet Connection."

55.5.3.3 Testing the Spaces Portlet Connection

Use this procedure to test the Spaces portlet connection by modifying content and checking that the modification shows up in the PeopleSoft application.

1. On the Spaces page that you added the PeopleSoft portlet to, modify some information that you can verify the changes for in the PeopleSoft application.
2. Save your changes and confirm that the changes also appear in the PeopleSoft application.

55.6 Integrating Oracle Business Intelligence Presentation Services

This section explains how to use JDeveloper to create and configure a WebCenter Portal application that integrates with the Oracle Business Intelligence Presentation Services catalog. At runtime, users can add business intelligence objects to their Framework application or Spaces pages.

This section includes the following subsections:

- Section 55.6.1, "Introduction to Integrating Oracle Business Intelligence Presentation Services"
- Section 55.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog"
- Section 55.6.3, "Integrating Oracle Business Intelligence Presentation Services in Framework Applications"
- Section 55.6.4, "Integrating Oracle Business Intelligence Objects in WebCenter Portal: Spaces"

55.6.1 Introduction to Integrating Oracle Business Intelligence Presentation Services

This section explains how you use JDeveloper to create a Framework application, or configure Spaces to integrate with the Oracle Business Intelligence Presentation Services catalog. For more information about adding Oracle Business Intelligence objects to Framework applications, see "Embedding Business Intelligence Objects in ADF Applications" in the *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*.

This section includes the following subsections:

- Section 55.6.1.1, "Understanding Oracle Business Intelligence Presentation Services Integration"
- Section 55.6.1.2, "Requirements for Integrating Oracle Business Intelligence Presentation Services"

55.6.1.1 Understanding Oracle Business Intelligence Presentation Services Integration

You can use JDeveloper to create Framework applications that integrate with Oracle Business Intelligence Presentation Services. At runtime, these applications include the Presentation Services catalog in the WebCenter Portal - Resource catalog. Users can then browse for and add business intelligence analyses, dashboard pages, dashboards, or scorecard components (strategy maps, strategy trees, KPI watchlists, cause and effect maps, and custom views) to their Framework application pages. Any filters, prompts, and actions links included in the Business Intelligence objects will work within the Framework application or Spaces pages.

At runtime, WebCenter Portal users can expand and browse the Presentation Services catalog's folders to view an analysis' views. The following view types display in the Presentation Services catalog: table, pivot table, chart, funnel chart, gauge, narrative, ticker and title. The following view types do not display in the Presentation Services catalog: view selector, column selector, logical SQL, and no-results view.

Users can also browse the dashboard folder for the pages associated with the dashboard; however, users cannot browse within the dashboard pages to see their components (for example, any analyses embedded in the dashboard). Users cannot include entire Scorecards (only a Scorecard's components) in their Framework application pages.

55.6.1.2 Requirements for Integrating Oracle Business Intelligence Presentation Services

To create a Framework application that integrates with Oracle Business Intelligence Presentation Services, you must have installed Oracle JDeveloper 11g Release 1 (11.1.1.6) or later and the required Oracle BI EE and WebCenter Portal extensions.

To properly create the Framework application, you must configure the library settings, update the `weblogic.xml` and `weblogic-application.xml` files, and have properly configured security.

For both Framework applications and Spaces, you must also set up a connection to the BI application as well as configuring security as described in "Creating an Oracle BI EE Presentation Services Connection" in the *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*. You will also need to specify the credentials for the connection, as described in Section 55.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog."

55.6.2 Configuring Credentials for Connecting to the Oracle BI Presentation Catalog

At design time, you need to specify credentials to connect to the Oracle BI Presentation Catalog. These credentials are used to retrieve the list of business intelligence objects (for example, analyses, dashboards, and scorecard components) from the Oracle BI Presentation Catalog.

This process ensures that the login to the Presentation Server is the same as the current user of the application and any access checks are performed as the current user, and data is fetched as the current user. If the ADF page contains business intelligence objects to which the user does not have access, the ADF page returns a message stating that the user does not have the proper permissions to access these objects.

Note that the **Perform impersonation** parameter should be set to `true` when security is enabled.

This section contains the following subsections:

- Section 55.6.2.1, "Creating the BIImpersonateUser"
- Section 55.6.2.2, "Granting Permissions to BIImpersonateUser"

55.6.2.1 Creating the BIImpersonateUser

Use the following procedures to create a BIImpersonateUser user to secure an application that uses an Oracle BI EE Presentation Services connection and includes Oracle BI EE objects. ADF security must be enabled for your application before you can apply the impersonate user credentials to the Oracle BI EE Presentation Services connection.

The Impersonate User feature secures applications that contain Oracle BI EE objects when Oracle BI EE and ADF are not sharing an Oracle Internet Directory (OID). Before you begin the process of creating and using Impersonate User, you must confirm that this capability is configured in your environment.

Before you perform this procedure, make sure that either you or the Administrator have created users in the WebLogic Server's Oracle BI EE realm and assigned the BIConsumer role to each user in this realm.

Follow the steps below to create the BIImpersonateUser user:

1. Open WebLogic Server for your Oracle BI EE instance.
2. Locate the Domain Structure pane and select Security Realm.

   The Realms pane displays.
3. In the Realms pane, select **<myrealm>**.

   The Settings dialog displays.
4. In the Settings dialog, open the Users and Groups tab.
5. Confirm that the Users tab is displaying and click **New**.
6. Enter `BIImpersonateUser` for the user name and enter a password.
7. Click **OK**.

55.6.2.2 Granting Permissions to BIImpersonateUser

Follow the steps below to use Fusion Middleware Control to grant permissions to BIImpersonateUser:

1. Open Fusion Middleware Control for your Oracle BI EE instance.

2. Locate and select `bifoundation_domain`.

   The bifoundation_domain pane displays.

3. In the `bifoundation_domain` pane, click the **WebLogic Domain** list, highlight `Security`, and select **Application Policies**.

   The Search pane displays.

4. In the **Select Application Name to Search** field's list, select **obi**.

5. Click **Create**.

   The Add Permission dialog displays.

6. In the **Permission Class** field, select `oracle.security.jps.ResourcePermission`. In the Resource Name field, select `oracle.bi.server.impersonateUser`.

7. Click **OK**.

8. In the Grantees pane, click **Add User**.

   The Add User dialog displays.

9. In the Available Users list, select `BIImpersonateUser` and click **Move** to move it to the Selected Users list.

10. Click **OK**.

11. In the `bifoundation_domain` pane, click **OK**.

12. If the changes that you made do not display, stop and restart the following servers:

    - Oracle BI EE Server
    - Oracle BI EE Presentation Server
    - WebLogic Server

55.6.3 Integrating Oracle Business Intelligence Presentation Services in Framework Applications

This section describes how to install the required BI extensions, and create or modify a Framework application that integrates with Oracle Business Intelligence Presentation Services.

This section includes the following subsections:

- Section 55.6.3.1, "Installing the Required Business Intelligence Extensions"
- Section 55.6.3.2, "Creating a Framework Application that Integrates with Oracle BI Presentation Services"
- Section 55.6.3.3, "Testing the Framework Application"
- Section 55.6.3.4, "Post-Deployment Connection Configuration"
- Section 55.6.3.5, "Securing Oracle BI EE Objects Included in a Framework Application"

55.6.3.1 Installing the Required Business Intelligence Extensions

Use this procedure to install the required Business Intelligence extensions:

1. From the JDeveloper Help menu, select **Check for Updates**.

2. Follow the steps in the wizard to locate, download, and install the Oracle BI EE extensions. Be sure to install the bundles in the following order:
   - `bi-soap-cn_bundle`
   - `bi-vo_bundle`
   - `bi-adf_bundle`
   - `bi-adf-taskflow-extension_bundle`
3. Also use the wizard to download and install the Oracle Webcenter Framework Extension bundle (`oracle.webcenter.framework_bundle.zip`).

55.6.3.2 Creating a Framework Application that Integrates with Oracle BI Presentation Services

Use the following procedures to create or modify a Framework application that can access the Oracle Business Intelligence Presentation Services catalog at runtime. Before following these steps you should already have created the BIImpersonateUser, as described in Section 55.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog."

This section includes the following subsections:

- Section 55.6.3.2.1, "Creating the Framework Application and JSF Page"
- Section 55.6.3.2.2, "Configuring the Application to Display the Oracle BI EE Presentation Catalog"
- Section 55.6.3.2.3, "Enabling the BI ADF Task Flow Permission"

55.6.3.2.1 Creating the Framework Application and JSF Page

Use the following procedure to create a Framework application and add a JSF page.

To create a Framework application and JSF page:

1. In Oracle JDeveloper, create a new `WebCenter Portal - Framework application` (be sure to check the checkbox for **Configure the Application with Standard Portal Features**).

   The user name/password in this case should be `BIImpersonateUser/<password>`, where `<password>` is the password that you set when you created the user.
2. Create a connection to the BI application as described in the section "How to Configure SSL and Create an Oracle BI EE Presentation Services Connection" in the *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*.

   When you create the connection, be sure to set **Perform impersonation when security is enabled** the to `true` and change the username and password to the Impersonate User credentials (`BIImpersonateUser/<password>`) as described above.
3. In the Navigation panel, right-click on the Portal project and select **New**.

   The New Gallery dialog displays.
4. In the **Categories** field's tree structure, locate and expand the Web Tier.

   A list with the descriptions of the available options display in the **Items** field.
5. From this list, select **JSF Page** and click **OK**.

The Create JSF Page dialog displays.

6. Create the JSF page, making sure that the `Create an XML Document (*.jspx)` field is selected. For more information about setting up the JSF page, click the Help button to access the online help.
7. Click **OK**.

   The newly created JSPX page displays.
8. In the Component Palette pane, select **Composer**.

   The field updates to display the corresponding components, layouts, and operations.
9. From the Common Components category, select **Change Mode Link** and drag and drop it onto your page.
10. From the Common Components category, select **Page Customizable**.
11. Save the application.
12. Right-click the page in Design view and choose **Edit Authorization**.
13. Navigate to **Granted to > Select authenticated-users** and under **Actions** grant authenticated users `Customize`, `Grant`, `Personalize`, and `View` privileges.

55.6.3.2.2 Configuring the Application to Display the Oracle BI EE Presentation Catalog

Use this procedure to add the Presentation Catalog to the Framework application's folder list. At runtime, the catalog displays in the WebCenter Portal - Resource Catalog where the users can browse to business intelligence objects and add them to their Framework application page.

To display the Presentation Catalog in the Framework application's folder list:

1. In the Framework application that you created with Oracle JDeveloper, from the Application menu, right-click **Portal Project** and select **Project Properties**.

   The Project Properties dialog displays.
2. In the available properties pane, select **Technology Scope** and from the Available field, select **Business Intelligence ADF View Components**, and click the **Add** (>) icon to add it to the project's technology scope. Note that you can also add the BI technology libraries when you create a new application.
3. In the available properties pane, select **Libraries and Classpath**, and click **Add Library**.

   The Add Library dialog displays.
4. Add the following libraries to the project, and click **OK**.
   - BI ADF Runtime
   - BI ADF Taskflow
   - BI SOAP Connection Runtime
   - BI View Object Runtime
5. Save the application.
6. In JDeveloper, open the default resource catalog file (`default-catalog.xml`) located in **Portal > Web Content > Oracle > webcenter > portalapp > catalogs**.
7. Add the following `<customFolder>` entry under the `<catalogDefinition> - ><contents>` folder:

```
<customFolder id="oracleBIContent"
    name="Oracle BI"
    description="Oracle BI Presentation Services Content"
    factoryClass="oracle.bi.presentation.rescat.
      BICatalogInitialContextFactory"/>
```

8. Edit `Weblogic.xml` and add references to `oracle.bi.adf.view.slib` and `oracle.bi.adf.webcenter.slib` as shown below:

```
   <library-ref>
      <library-name>oracle.bi.adf.view.slib</library-name>
   </library-ref>
   <library-ref>
      <library-name>oracle.bi.adf.webcenter.slib</library-name>
   </library-ref>
</weblogic-web-app>
```

9. Edit `weblogic-application.xml` and add a reference to `oracle.bi.adf.model.slib` as shown below:

```
   library-ref>
     <library-name>oracle.bi.adf.model.slib</library-name>
   </library-ref>
</weblogic-application>
```

55.6.3.2.3 Enabling the BI ADF Task Flow Permission

Use the following procedure to enable the BI ADF task flow permissions in your application's security settings.

To enable the BI ADF task flow permissions:

1. Open the `jazn-data.xml` file from **Application Resources Pane > META-INF > jazn-data.xml**.
2. Locate the following code in the file:

```
<grantee>
    <principals>
        <principal>
            <class>
              oracle.security.jps.internal.core.principals.JpsAnonymousRoleImpl
             </class>
            <name>anonymous-role</name>
        </principal>
    </principals>
</grantee>
```

3. Add the following element:

```
<permission>
    <class>oracle.adf.controller.security.TaskFlowPermission</class>
    <name>/WEB-INF/oracle/bi/presentation/taskflow/
     task-flow-definition.xml#task-flow-definition</name>
    <actions>customize,personalize,view</actions>
</permission>
```

   For example:

```
<grant>
   <grantee>
      <principals>
         <principal>
```

```
            <class>oracle.security.jps.internal.core.principals
            .JpsAnonymousRoleImpl</class>
            <name>anonymous-role</name>
          </principal>
        </principals>
      </grantee>
    <permissions>
      <permission>
          <class>oracle.adf.controller.security.TaskFlowPermission</class>
          <name>/WEB-INF/oracle/bi/presentation/taskflow/
           task-flow-definition.xml#task-flow-definition</name>
          <actions>customize,personalize,view</actions>
      </permission>
```

55.6.3.3 Testing the Framework Application

This section describes how to test the Framework application by performing the following procedures:

- Adding Oracle BI Presentation Services catalog content and modifying a report or dashboard's prompt values
- Confirming and modifying a business intelligence object's initialization parameters.

This section includes the following subsections:

- Section 55.6.3.3.1, "Adding Oracle BI Objects to the Resource Catalog"
- Section 55.6.3.3.2, "Adding Business Intelligence Content at Runtime"
- Section 55.6.3.3.3, "Modifying a Business Intelligence Object's Prompt Values"
- Section 55.6.3.3.4, "Modifying a Business Intelligence Task Flow's Initialization Parameters"

55.6.3.3.1 Adding Oracle BI Objects to the Resource Catalog

Before you can add Oracle BI content to a Framework application page, you must add objects stored in the Oracle BI Presentation Catalog to the WebCenter Portal - Resource Catalog.

1. Run the application in a browser.
2. Log into the Framework application as an administrator.
3. Click **Administration**.

   The WebCenter Portal Administration Console displays.
4. Open the **Resources** tab.
5. Click **Resources Catalogs**.
6. Click **Create a Resource Catalog**.
7. In the **Name** field, enter the name of the catalog you are creating. Complete the other fields on this dialog, as necessary.
8. Click **Edit > Show** for the resource catalog.
9. Click the **Edit** menu and then choose **Edit**.
10. From the **Add** menu, choose **Add From Library...**.
11. Double-click **Connections**.

The BI Presentation Services folder displays in the right frame. Open this folder to display the Oracle BI objects.

12. Browse to and select the objects that you want to add.
13. Click **Add** to add the selected objects to the catalog.
14. Open the Configuration tab and select the resource catalog you just created as the **Default Resource Catalog**.

55.6.3.3.2 Adding Business Intelligence Content at Runtime

Use this procedure to test the Framework application by adding business intelligence content to the Framework application.

1. Run the `.jspx` page that you created.
2. In the running page, click the **Edit** link to enter the edit mode.
3. Click **Add Content**.

   The WebCenter Portal - Resource Catalog displays.
4. Browse to and open the Oracle BI folder.
5. Click the host machine to which you want to connect.

   The contents of the catalog display.
6. Browse to and click the **Add** link for the object that you want to add to the `.jspx` page.
7. Confirm that the application correctly added the object.

55.6.3.3.3 Modifying a Business Intelligence Object's Prompt Values

Use this procedure to test the Framework application by changing an analysis or dashboard's filter or prompt values.

1. Run the `.jspx` page that you created.
2. In the running page, click the **Edit** link to enter the edit mode.
3. Add an analysis or dashboard that contains a filter or prompt.

   For more information about adding business intelligence content to the .jspx page, see Section 55.6.3.3.2, "Adding Business Intelligence Content at Runtime."
4. Click **Parameters**.

   The Editing Page dialog displays.
5. Modify the values and click **Save**.
6. Confirm that the application correctly applied the modified prompt values.

55.6.3.3.4 Modifying a Business Intelligence Task Flow's Initialization Parameters

Use the following procedure to test the business intelligence task flow's initialization parameters.

1. Run the `.jspx` page that you created.
2. In the running page, click the **Edit** link to enter the edit mode.
3. Add a business intelligence object.

   For more information about adding business intelligence content to the .jspx page, see Section 55.6.3.3.2, "Adding Business Intelligence Content at Runtime."

4. Locate the business intelligence object and click the **Edit** (wrench) icon.

   The Component Properties dialog displays.

5. Open the Parameters tab, modify the object's parameters, and click **OK**.

6. Confirm that the application correctly applied the modified prompt values.

55.6.3.4 Post-Deployment Connection Configuration

After an application has been deployed, use Fusion Middleware Control to add a new Oracle BI EE Presentation Services connection to the application, or modify the application's existing Oracle BI EE Presentation Services connection with the user name and password for BIImpersonateUser.

For more information about modifying these connections, see "Adding or Modifying a Presentation Services Connection After Deployment" in the *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*.

> **Note:** If the Framework application and the Oracle Business Intelligence application do not share the same identity store, you must create the relevant users in both system.

55.6.3.5 Securing Oracle BI EE Objects Included in a Framework Application

Impersonate User lets you secure a Framework application that integrates with Oracle BI EE Presentation Services. For more information about Impersonate User and how to implement it to secure your application, see "How to Create and Use Impersonate User" in the *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*.

55.6.4 Integrating Oracle Business Intelligence Objects in WebCenter Portal: Spaces

Use the following procedures to test the WebCenter Portal: Spaces integration with the BI objects.

- Section 55.6.4.1, "Adding or Modifying a Presentation Services Connection After Deployment"
- Section 55.6.4.2, "Adding Oracle BI Objects to the Spaces Resource Catalog"
- Section 55.6.4.3, "Adding Oracle BI Content at Runtime"
- Section 55.6.4.4, "Modifying a Business Intelligence Object's Prompt Values"
- Section 55.6.4.5, "Modifying a Business Intelligence Task Flow's Initialization Parameters"

55.6.4.1 Adding or Modifying a Presentation Services Connection After Deployment

Before you can begin integrating BI objects in Spaces, you must first configure a connection from Spaces to the BI server. Oracle BI EE provides an ADF MBean that allows you to add a new connection to a deployed BI ADF or WebCenter Portal: Spaces application. You can also modify a deployed application's existing connection. MBeans are deployed with the application and can be accessed post-deployment using Fusion Middleware Control.

Prior to following the steps below, you should already have followed the steps in Section 55.6.2, "Configuring Credentials for Connecting to the Oracle BI Presentation Catalog" to specify credentials to connect to the Oracle BI Presentation Catalog.

> **Note:** If the Spaces application and the Oracle Business Intelligence application do not share the same identity store, you must create the relevant users in both system.

Follow the steps below to configure the connection after the application was deployed.

1. Open Fusion Middleware Control and select your WebLogic domain by doing one of the following:
   - If your application is deployed in the BI ADF domain, select the **Application Deployments <your domain>** tree node.
   - If your application is deployed in the Spaces domain, select the **WebCenter** tree node, and then the **Spaces** tree node, and then the **webcenter(11.1.1.4.0)** (**WC_Spaces**) tree node.
2. From the list, select **System MBean Browser**.

   The System MBean Browser pane displays.
3. In the System MBean Browser pane, navigate to the ADF Connections tree node by following the below path:
   a. Select the **Application Defined MBeans** tree node.
   b. Select the **oracle.adf.share.connections** tree node.
   c. Select the **Server: <my server name>** tree node.

      For example, `Server:DefaultServer` or `Server:WC_Spaces`.
   d. Select the **Application:<your application's name>** tree node.

      For example, `Application:Application2` or `Application:webcenter`.
   e. Select the ADF Connections tree node.
   f. Select the ADF Connections tree node.

      The corresponding MBean information displays in the Application Defined MBean pane.
4. In the Application Defined MBean pane, open the Operations tab and then click **createConnection/findOrCreateConnection** to create a Presentation Services connection.

   The Operation:createConnection dialog displays.
5. Specify the required values for the connection.

   In the connection type **Value** field, enter `BISoapConnection`, enter a name in the connection name **Value** field and click **Invoke** to create the connection.
6. In the System MBean Browser pane, click **Refresh** to refresh the tree so that the new connection displays.
7. To modify the connection, locate it in the System MBean Browser pane and click it.

   The connection's information displays in the Application Defined MBean pane.

8. Navigate to ADF Connections from the System MBean Browser pane and open the Attributes tab.
9. Enter the `BISoap` connection information, and then click **Apply** to apply your changes.
10. Open the Operations tab and click **Save** to save the connection.
11. When you click **Invoke**, you should get the following message "Confirmation Operation executed successfully."

55.6.4.2 Adding Oracle BI Objects to the Spaces Resource Catalog

Before you can add Oracle BI content to a Spaces page, you must add objects stored in the Oracle BI Presentation Catalog to the Spaces Resource Catalog.

1. Log into Spaces as an administrator.
2. In Spaces, click **Administration**.

   The Spaces Administration pages displays.
3. Open the **Resources** tab.
4. Click **Resources Catalogs**.
5. Click **Create a Resource Catalog**.
6. In the **Name** field, enter the name of the catalog you are creating. Complete the other fields on this dialog, as necessary.
7. Make the Resource Catalog available by choosing **Edit > Show**.
8. From the **Edit** menu, choose **Edit**.
9. Select the **Add** menu and then choose **Add From Library...**..
10. Double-click **Connections**.

    The BI Presentation Services folder displays.
11. Open this folder to display the Oracle BI objects and browse to and select the objects that you want to add.
12. Click **Add** to add the selected objects to the catalog.

55.6.4.3 Adding Oracle BI Content at Runtime

Use this procedure to create a WebCenter Portal: Spaces page and add Oracle BI objects to it. Before you perform this procedure, you must have added Oracle BI objects to the WebCenter Portal: Spaces resource catalog.

1. Log into Spaces and create a new space or access an existing space.
2. Make the Resource Catalog the default catalog for the space.
   a. Navigate to **Space Settings > General** and select the catalog created in Section 55.6.4.2, "Adding Oracle BI Objects to the Spaces Resource Catalog" as the default catalog for **Resource Catalog for Pages**.
   b. Click **Apply**.
   c. Click **Back to Space**.
3. Create a new page.
4. In the page, click **Add Content**.
5. Browse to the folder containing the Oracle BI objects.

6. Select an analysis or dashboard and click **Add**.

   The object that you selected is added to the page.

7. Click **Close** to exit the Add Content dialog.

55.6.4.4 Modifying a Business Intelligence Object's Prompt Values

Use this procedure to test the WebCenter Portal: Spaces page by changing an analysis or dashboard's filter or prompt values.

1. Open the page that you created.
2. In the running page, click **Page Actions** and then click the **Edit** link to enter edit mode.
3. Add an analysis or dashboard that contains a filter or prompt. For more information about adding Oracle BI objects to the page, see Section 55.6.4.3, "Adding Oracle BI Content at Runtime."
4. Without exiting the edit mode of the page, save the Spaces page.
5. In the Spaces page, modify the prompt values and click **OK**.
6. Exit edit mode, save the page and confirm that the application correctly applied the prompt values.

55.6.4.5 Modifying a Business Intelligence Task Flow's Initialization Parameters

Use the following procedure to test the business intelligence task flow's initialization parameters.

1. Open the page that you created.
2. In the running page, click **Page Actions** and then click the **Edit** link to enter the edit mode.
3. Add an analysis or dashboard that is part of a task flow. For more information about adding business intelligence content to the `.jspx` page, see Section 55.6.4.3, "Adding Oracle BI Content at Runtime."
4. Without exiting the edit mode for the page, save the Spaces page.
5. Locate the business intelligence object and click the **Edit** (wrench) icon.

   The Component Properties dialog displays.

6. On the Spaces page, open the **Parameters** tab and modify the object's parameters and click **OK**.
7. Open the Parameters tab, modify the object's parameters, and click **OK**.
8. Exit edit mode, save the page and confirm that the application correctly applied the modified parameter values.

56

Extending the Spaces Application Using JDeveloper

WebCenter Portal's Spaces application can be customized using browser-based tools and also using JDeveloper. This chapter describes how to customize the Spaces application and Spaces resources in JDeveloper and subsequently utilize them in the Spaces application. For example, you may want to deploy additional shared libraries that include custom code or some additional task flows. Or, you might want to edit Spaces resources, such as skins, page templates, and navigations, in JDeveloper.

It includes the following sections:

- Section 56.1, "Downloading a Workspace for Spaces Development"
- Section 56.2, "Including Additional Shared Libraries in Spaces"
- Section 56.3, "Developing Resources for the Spaces Application"

Audience

This section is intended for developers who have basic knowledge of JDeveloper with WebCenter Portal: Framework extensions and want to perform advanced customizations for their Spaces deployment.

56.1 Downloading a Workspace for Spaces Development

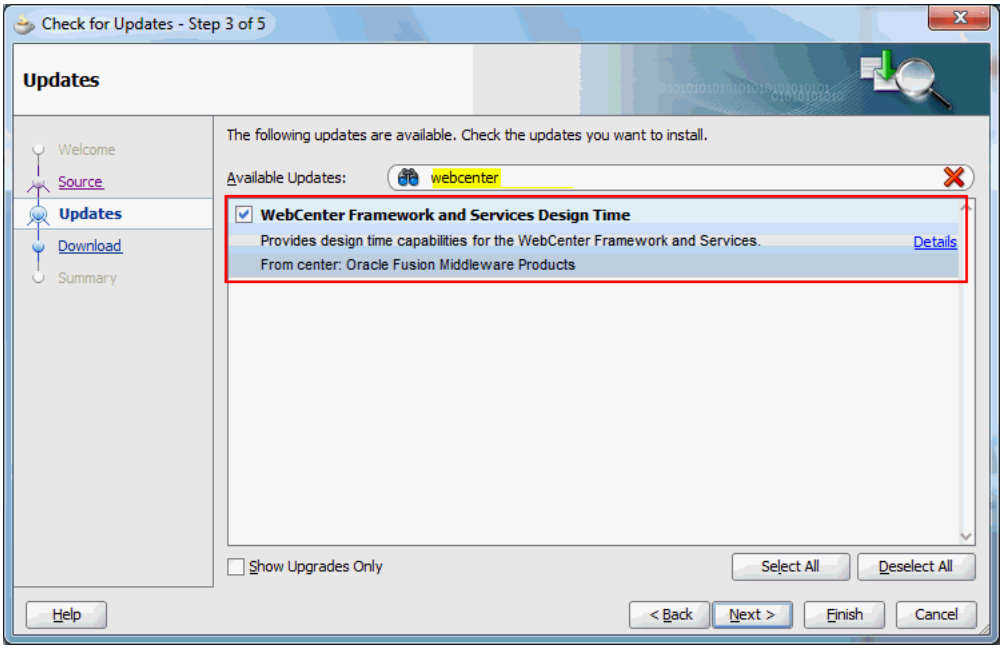
A JDeveloper workspace for developing Spaces extensions is available as a download from Oracle Technology Network - this workspace is named **DesignWebCenterSpaces.jws**.

To download the Spaces development workspace:

1. Ensure the Spaces application is installed, configured, and working as expected.
2. Download and install Oracle JDeveloper 11g (11.1.1.6.0).

   Oracle JDeveloper 11g (11.1.1.6.0) is available for download from: `http://www.oracle.com/technetwork/developer-tools/jdev/downloads/index.html`
3. Download and install Oracle JDeveloper WebCenter Portal Framework and Services Design Time Extensions:
   a. In JDeveloper, select **Help** > **Check for Updates**.
   b. Click **Next**, and then **Next** again to display all available updates.
   c. Use the search field to filter the list, for example, enter '`webcenter`'.

Figure 56–1 Download Oracle JDeveloper WebCenter Portal Framework and Services Design Time Extensions



d. Select **WebCenter Framework and Services Design Time**.

e. Click **Finish**.

See also, Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper."

4. Download the Spaces development ZIP file (`DesignWebCenterSpaces_PS5.zip`) from Oracle Technology Network:

`http://download.oracle.com/otndocs/tech/webcenter/files/DesignWebCenterSpaces_PS5.zip`

5. Unzip the content locally (Table 56–1).

Table 56–1 DesignWebCenterSpaces_PS5.zip - Unzip Folders

Unzip Folders	Description
\DesignWebCenterSpaces	Includes `DesignWebCenterSpaces.jws`. This workspace enables you to configure your shared library list, that is, you can specify one or more shared libraries containing various customizations (such as custom java code, custom task flows, and so on) that you want to utilize in the Spaces application. The workspace also provides a design environment for editing and uploading Spaces resources.
\copy_to_common	Includes a `common` directory that contains WebCenter Portal WLST commands. WebCenter Portal's WLST commands are required to upload customized resources directly to the Spaces application from `DesignWebCenterSpaces.jws`.

Table 56–1 (Cont.) DesignWebCenterSpaces_PS5.zip - Unzip Folders

Unzip Folders	Description
\copy_to_jdev_ext	Includes `oracle.webcenter.portal.jar`. Required update to Oracle JDeveloper WebCenter Portal Framework and Services Design Time Extensions.

6. Copy WebCenter Portal's WLST (WebLogic Scripting Tool) commands to the Oracle home directory where JDeveloper is installed:
   a. Navigate to: `<DesignWebCenterSpaces_Unzip_Location>\copy_to_common`
   b. Make a copy of the `common` directory.

      The `common` directory is located under `<DesignWebCenterSpaces_Unzip_Location>\copy_to_common`.
   c. Navigate to: `<JDeveloper_install_directory>\oracle_common`

      Tip: The `JDeveloper_install_directory` is the base directory under which JDeveloper is also installed, that is, the same directory that you configure as the `jdeveloper.install.home.directory` in `config.properties` (see also Table 56–2).
   d. Copy the content of the `common` directory here.

      Tip: On Linux, use the following command: `cp -r <DesignWebCenterSpaces_Unzip_location>/copy_to_common/common/.`
7. Copy new version of `oracle.webcenter.portal.jar` to the Oracle JDeveloper extensions directory:
   a. Navigate to: `<DesignWebCenterSpaces_Unzip_Location>\copy_to_jdev_ext`
   b. Make a copy of `oracle.webcenter.portal.jar`.

      The `oracle.webcenter.portal.jar file` is located under `<DesignWebCenterSpaces_Unzip_Location>\copy_to_jdev_ext`.
   c. Navigate to: `<JDeveloper_install_directory>\jdeveloper\jdev\extensions\`

      Tip: The `JDeveloper_install_directory` is the base directory under which JDeveloper is also installed, that is, the same directory that you configure as the `jdeveloper.install.home.directory` in `config.properties` (see also Table 56–2).
   d. Copy the `oracle.webcenter.portal.jar file` here.
8. Restart JDeveloper if it is open.

56.2 Including Additional Shared Libraries in Spaces

This section includes the following subsections:

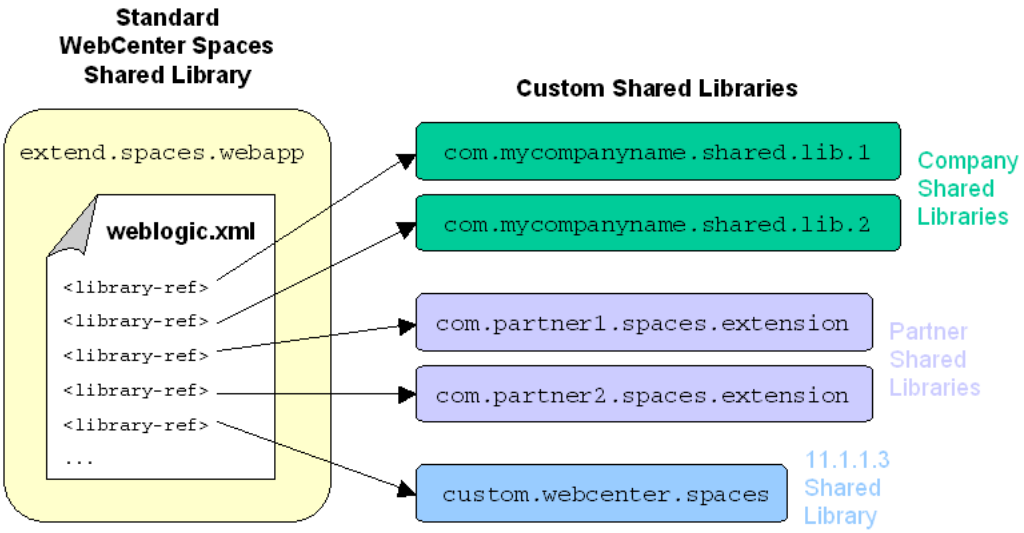
- Section 56.2.1, "What You Should Know About Shared Library Deployment for Spaces"
- Section 56.2.2, "Deploying Your Own Custom Code and Task Flows in Shared Libraries"
- Section 56.2.3, "Rebuilding the Spaces Shared Library List"

56.2.1 What You Should Know About Shared Library Deployment for Spaces

The Spaces application includes the standard shared library `extend.spaces.webapp.war`. This .WAR file includes a deployment descriptor (`weblogic.xml`) which can reference other libraries that contain Spaces customizations.

Therefore, if you have custom code or task flows deployed in several shared libraries from multiple sources you can add them to the `extend.spaces.webapp` shared library list, as illustrated in Figure 56–2.

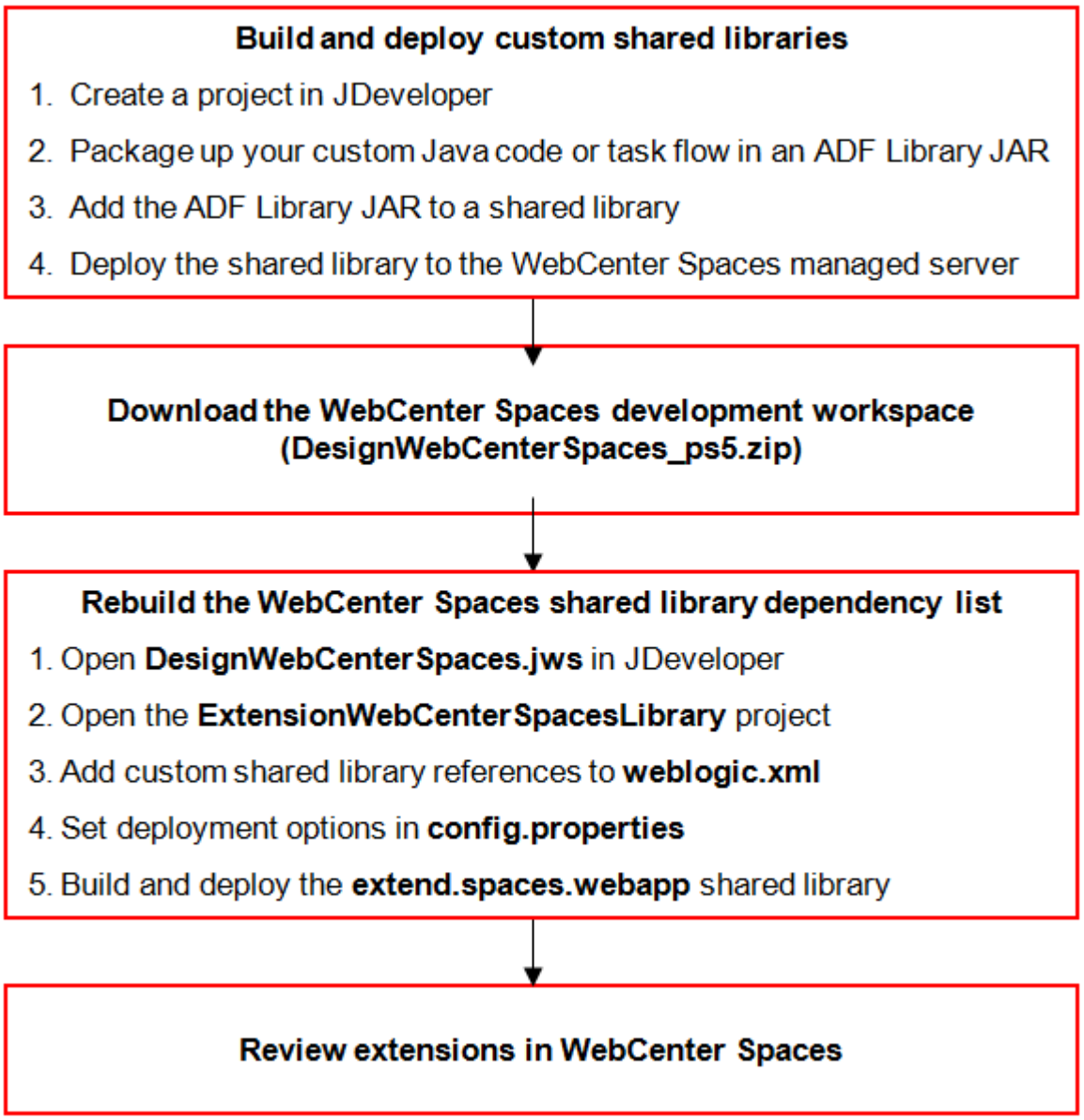
Figure 56–2 Referencing Custom Shared Libraries in extend.spaces.webapp.war



This development model provides an easy way to utilize additional shared libraries in Spaces from multiple contributors, including developers, customers, partners.

Whenever you deploy a new shared library that includes Spaces extensions you must modify the Spaces shared library dependency list and redeploy `extend.spaces.webapp.war`. Figure 56–3 illustrates the process.

Figure 56–3 Overview - Including Additional Shared Libraries in Spaces



For more detail, see Section 56.2.2, "Deploying Your Own Custom Code and Task Flows in Shared Libraries" and Section 56.2.3, "Rebuilding the Spaces Shared Library List".

56.2.2 Deploying Your Own Custom Code and Task Flows in Shared Libraries

Developers can build extensions for the Spaces application, package them in ADF Library JARs, and then deploy them as shared libraries for use in Spaces. For example, you can:

- Develop custom code in JDeveloper and make it available in the Spaces application.
- Develop custom task flows in JDeveloper and make them available to the Spaces application

To help you get started, Oracle provides a sample workspace in which you can develop extensions, and build and deploy Spaces shared libraries that contain your custom code.

The sample workspace, `SampleWebCenterSpacesExtension.jws`, contains sample extensions and scripts that enable you to deploy custom code for Spaces. For more information, download the accompanying whitepaper, *Using Spaces Extension Samples (11.1.1.6.0)*, from Oracle Technology Network at `http://www.oracle.com/technetwork/index.htm`.

56.2.3 Rebuilding the Spaces Shared Library List

Whenever you deploy a new shared library that includes Spaces extensions you must update the Spaces shared library dependency list and redeploy `extend.spaces.webapp.war`.

This section describes:

- Editing the Shared Library List
- Setting Build and Deployment Options (config.properties)
- Building and Deploying the Shared Library List (extend.spaces.webapp.war)
- Troubleshooting Shared Library Deployment
- Reverting to a Previous Shared Library List

56.2.3.1 Editing the Shared Library List

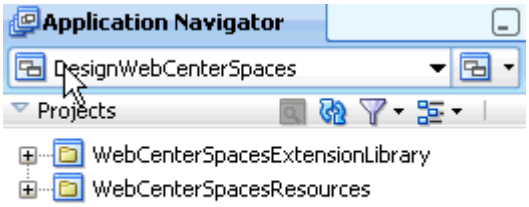
Use the *WebCenterSpacesExtensionLibrary* project to build and deploy `extend.spaces.webapp.war`—a shared library containing `weblogic.xml` in which you configure the Spaces shared library dependency list.

1. Download and unzip the Spaces development workspace.

   See Section 56.1, "Downloading a Workspace for Spaces Development".

2. Navigate to the unzip directory, and open `DesignWebCenterSpaces\DesignWebCenterSpaces.jws` in JDeveloper (Figure 56–4).

Figure 56–4 Projects in the DesignWebCenterSpaces Workspace

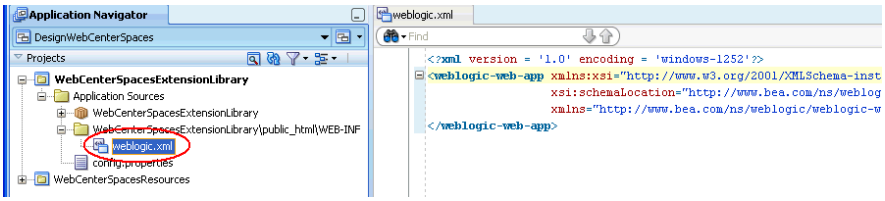


3. Expand the **WebCenterSpacesExtensionLibrary** project.

4. Navigate to **weblogic.xml**, under **Resources\WebCenterSpacesExtensionLibrary\public_html\WEB-INF** (Figure 56–5).

   Out-of-the-box, this file is blank, that is, no additional shared libraries are listed in the file.

Figure 56–5 WebCenterSpacesExtensionLibrary - Out-of-the-box weblogic.xml

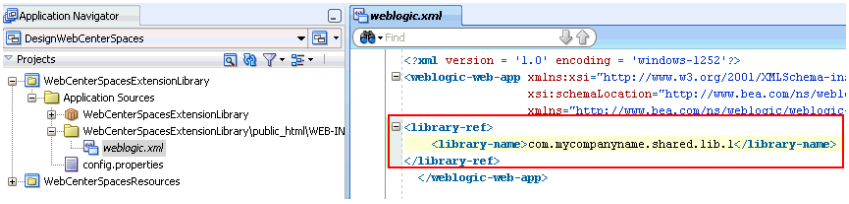


5. Add a `<library-ref>` entry for each shared library that you want to use in Spaces.

   For example, you can include a single library entry, as shown in Figure 56–6:

```
<library-ref>
  <library-name>com.mycompanyname.shared.lib.1</library-name>
</library-ref>
```

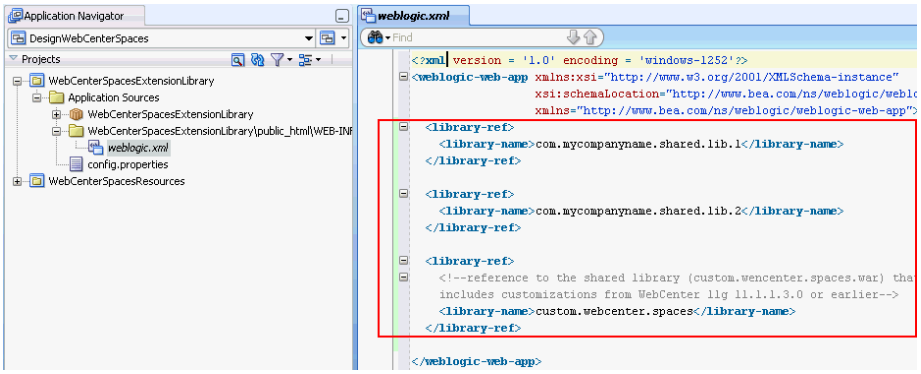
Figure 56–6 weblogic.xml - Single Shared Library Reference



   Or, reference multiple shared libraries, as shown in Figure 56–7:

```
<library-ref>
    <library-name>com.mycompanyname.shared.lib.1</library-name>
</library-ref>
<library-ref>
    <library-name>com.mycompanyname.shared.lib.2</library-name>
</library-ref>
<library-ref>
     <!--reference to the shared library (custom.wencenter.spaces.war) that
      includes customizations from WebCenter 11gR1 11.1.1.3.0 or earlier-->
    <library-name>custom.webcenter.spaces</library-name>
</library-ref>
```

Figure 56–7 weblogic.xml - Multiple Shared Library References



Note: If you want to use custom extensions developed for a previous WebCenter Spaces release (11.1.1.3.0 or earlier), include the name of the old shared library, **`custom.webcenter.spaces`**, in `weblogic.xml`, as shown in Figure 56–7.

6. Ensure each shared library that you reference is deployed on the Spaces managed server.

The next section describes how to set build and deployment options for the shared library list (`extend.spaces.webapp.war`).

56.2.3.2 Setting Build and Deployment Options (config.properties)

Before you can build the shared library list and deploy `extend.spaces.webapp.war` to the Spaces managed server, you must provide some information about your environment and your WebCenter Portal installation in the configuration file `config.properties`.

To set build and deployment properties for the `extend.spaces.webapp` shared library:

1. Open **DesignWebCenterSpaces.jws**.

   See also, Section 56.1, "Downloading a Workspace for Spaces Development"

2. Expand the **WebCenterSpacesExtensionLibrary** project.
3. Open **config.properties**.
4. Enter information about your JDeveloper environment and WebCenter Portal installation, as shown in Table 56–2.

   The `config.properties` file describes each property and offers examples. The defaults provided are only samples and must be replaced with your own, installation-specific values.

Table 56–2 Configuring config.properties to Enable Deployment to the Managed Server

Configuration Property	Description
`jdeveloper.install.home.directory`	Base directory where JDeveloper is installed. The directory you specify contains other folders such as `jdeveloper`, `wlserver_10.3`, `modules`, and so on.
`wlst.executable`	Name of the WLST executable file. Either `wlst.cmd` (on Windows) or `wlst.sh` (on Linux).
`extending.spaces.home.dir`	Path to the Spaces development workspace directory. For example, `C:\DesignWebCenterSpaces`
`oracle.jdeveloper.ojdeploy.path`	Path to `ojdeploy.exe` (Windows) or `ojdeploy` (Linux).
`wls.port`	Port number on which the WLS Administration Console is running
`wls.host`	Host machine on which the Spaces application is running and where the custom shared library is to be deployed
`wls.userkey`	Name and location of the file storing administrator passwords. `config.properties` describes how to generate the file using WLST. The file can be copied to any suitable, accessible location.
`wls.userconfig`	Name and location of the file storing administrator user details. `config.properties` describes how to generate the file using WLST. The file can be copied to any suitable, accessible location.

Table 56–2 (Cont.) Configuring config.properties to Enable Deployment to the Managed

Configuration Property	Description
`wls.target`	Targets on which the Spaces shared library is to be deployed. For example, `WC_Spaces`.
`wls.webcenter.app.target`	Targets on which the `webcenter.ear` file is to be deployed. For example, `WC_Spaces`.
`webcenter.app.name`	Name of the Spaces application. Always `webcenter`.
`restart.implementation.version.suffix`	Controls incremental implementation version numbers.

5. Save your updates to `config.properties`.

Updates to `config.properties` are available immediately so there is no need to restart JDeveloper.

The next section describes how to build and deploy the shared library list `extend.spaces.webapp.war`.

56.2.3.3 Building and Deploying the Shared Library List (extend.spaces.webapp.war)

You must rebuild and redeploy the `extend.spaces.webapp.war` shared library whenever you modify the shared library list (`weblogic.xml`).

To build and deploy the shared library list:

1. Open `DesignWebCenterSpaces.jws` in JDeveloper.

   See also, Section 56.1, "Downloading a Workspace for Spaces Development".

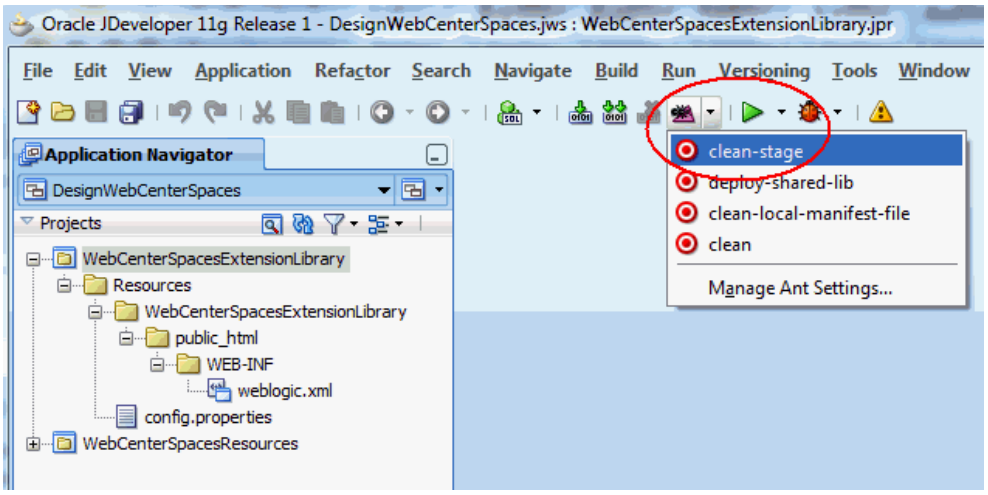
2. Expand the **WebCenterSpacesExtensionLibrary** project.

3. If you have not done so already, specify environment details in `config.properties`.

   See also, Section 56.2.3.2, "Setting Build and Deployment Options (config.properties)".

4. To rebuild `extend.spaces.webapp.war`, open the **Run Ant** dropdown menu, and then select **clean-stage** (Figure 56–8)

Figure 56–8 Building a Customized WebCenter Portal: Spaces Extension Library



This generates a new version of the shared library list `extend.spaces.webapp.war`. The implementation version number associated with the new .WAR is saved to `MANIFEST.MF` as follows:

`<UnzipDir>/DesignWebCenterSpaces/WebCenterSpacesExtensionLibrary/META-INF/MANIFEST.MF`

5. To redeploy the Spaces shared library list, open the **Run Ant** dropdown menu, and then select **deploy-shared-lib**.

A new version of the shared library list is deployed to the Spaces managed server. To verify the new deployment, login to the WLS Administration Console, navigate to the deployment overview page, and check the implementation version, that is, navigate to:

Deployments> extend.spaces.webapp>Overview

If the latest version is not active, refer to the Section 56.2.3.5, "Troubleshooting Shared Library Deployment".

56.2.3.4 Reverting to a Previous Shared Library List

If there is a problem with the latest shared library list or you want to revert to a previous version for some reason, you can undeploy (remove) the current version and revert to the previous version, using the WLS Administration Console.

You can remove unwanted shared library versions too. If you go through several "change-build-deploy-test" iterations, each incremental version is retained by default. As the Spaces application only uses the latest shared library version you can clean up or delete previous versions if you want.

Before undeploying the latest version, you must shut down the managed server on which the Spaces application is running. Once you have removed the latest version, you can restart the managed server.

Note: Oracle recommends that you do not delete the original `extend.spaces.webapp` shared library (version 11.1.1) as this enables you to revert to the out-the-box version if necessary.

1. Run the following script to stop the managed server on which the Spaces application is deployed:
   - UNIX: DOMAIN_HOME/bin/stopManagedWeblogic.sh

- Windows: DOMAIN_HOME\bin\stopManagedWeblogic.cmd

2. Login to the WLS Administration Console, navigate to the deployment overview page (**Deployments> extend.spaces.webapp>Overview**), and remove the latest shared library version.

   To revert to the out-of-the-box Spaces deployment, revert to the original `extend.spaces.webapp.war` shared library, that is, delete all other Spaces shared library versions, except for `extend.spaces.webapp.war` version 11.1.1.

3. Run the following script to start the managed server on which the Spaces application is deployed:

   - UNIX: DOMAIN_HOME/bin/startManagedWeblogic.sh
   - Windows: DOMAIN_HOME\bin\startManagedWeblogic.cmd

56.2.3.5 Troubleshooting Shared Library Deployment

- **Changes are not available after deployment even though deployment successful**.

  The Spaces application always uses the latest shared library version. Check that the implementation version in MANIFEST.MF matches the implementation version displayed in the WLS Administration Console.

  For example, check the value in `<UnzipDir>/DesignWebCenterSpaces/WebCenterSpacesExtensionLibrary/META-INF/MANIFEST.MF` is the same as that displayed in the WLS Admin Console under **Deployments> extend.spaces.webapp>Overview**

- **"Security: 090219" error displays:**

  ```
  weblogic.security.internal.encryption.EncryptionServiceException:
  weblogic.security.internal.encryption.EncryptionServiceException:
  weblogic.security.internal.encryption.EncryptionServiceException:
  [Security:090219]Error decrypting Secret Key com.rsa.jsafe.JSAFE_
  InputException: Invalid input length for decryption. Should be a multiple of
  the block size - 8
  ```

  Check that `wls.userconfig` and `wls.userkey` are both set correctly in the `config.properties` file, and verify that both security files specified are accessible

- **"DeployerException: Task 9 failed" displays:**

  ```
  weblogic.Deployer$DeployerException:
  weblogic.deploy.api.tools.deployer.DeployerException: Task 9 failed:
  [Deployer:149117]deploy library custom.webcenter.spaces
  [LibSpecVersion=11.1.1.2,LibImplVersion=11.1.1.2.5] on AdminServer,WC_Spaces.
  ```

  This error occurs if the implementation version of the new deployment and the existing deployment are the same. Use the WLS Administration Console to verify the current implementation version and then change the restart.implementation.version.suffix property in `config.properties` to this value. When you rebuild and redeploy the shared library, the implementation version should increment by '1'.

- **"java.lang.IllegalArgumentException" displays**:

  Restart the managed server on which the Spaces application is deployed.

- **Deployment hangs:**

If you encounter issues during the deployment phase (**Run Ant Target > deploy-shared-lib**) and the process hangs, end/kill the Ant process from inside JDeveloper, and then restart the Administration Server and the managed server for the Spaces application. If you encounter further issues restarting either the Administration Server or the managed server, end/kill the associated Java processes and restart both servers.

- **Newly deployed Spaces shared library is not active:**

  If a newly deployed shared library is not active, use WLS Administration Console to manually delete the non-active shared library version, and then rebuild (**Run Ant Target>clean-dist**) and redeploy (**Run Ant Target > deploy-shared-lib**) the Spaces shared library.

- **"Error occurred attempting to display the list of pages you can access. Please contact the administrator" displays after logging in:**

  Restart the managed server on which the Spaces application is deployed.

- **"WCS#2009.12.22.00.49.35: Fatal error occurred while getting members" displays:**

  Restart Spaces.

56.3 Developing Resources for the Spaces Application

This section describes the WebCenterSpacesResources project—-a design environment for Spaces resource development. Through this project, you can upload resources, developed or customized through JDeveloper, to the Spaces application.

This section includes the following subsections:

- Section 56.3.1, "Using the WebCenterSpacesResources Project for Round-Trip Development"
- Section 56.3.2, "Opening and Exploring the WebCenterSpacesResources Project"
- Section 56.3.3, "Enabling Direct Uploads to the Spaces Application"
- Section 56.3.4, "Editing Spaces Resources in JDeveloper"
- Section 56.3.5, "Uploading Resources Directly to the Spaces Application"
- Section 56.3.6, "Exporting Spaces Resources to an Archive"
- Section 56.3.7, "Troubleshooting Resource Round-Trip Development"

56.3.1 Using the WebCenterSpacesResources Project for Round-Trip Development

WebCenter Portal's round-trip development features provide a simple, convenient way to modify Spaces resources without redeploying the entire application. Round-trip development refers to features and techniques that allow you to download resources from a deployed application and upload them to JDeveloper for maintenance or enhancement. Once modified, you can upload the resource back to the Spaces application for immediate use or for testing.

Out-of-the-box, Spaces provides some default resources, such as skins and page templates, for people to use or modify. To keep Spaces easy to use, the built-in functionality for creating and editing resources within Spaces is purposely somewhat limited. If your resource requirements extend beyond the editing capabilities of the Spaces application, you can further develop resources using a JDeveloper project that is especially designed for that purpose—*WebCenterSpacesResources*.

Oracle recommends that you use this project to build and manage custom skins, page templates, page styles, navigations, content display templates, mashup styles, and resource catalogs. The project provides some sample resources, as well as other files and folders you might need. You do not have to develop resources from scratch; you can export existing resources from the Spaces application and edit them in JDeveloper, or use one of the samples as your starting point.

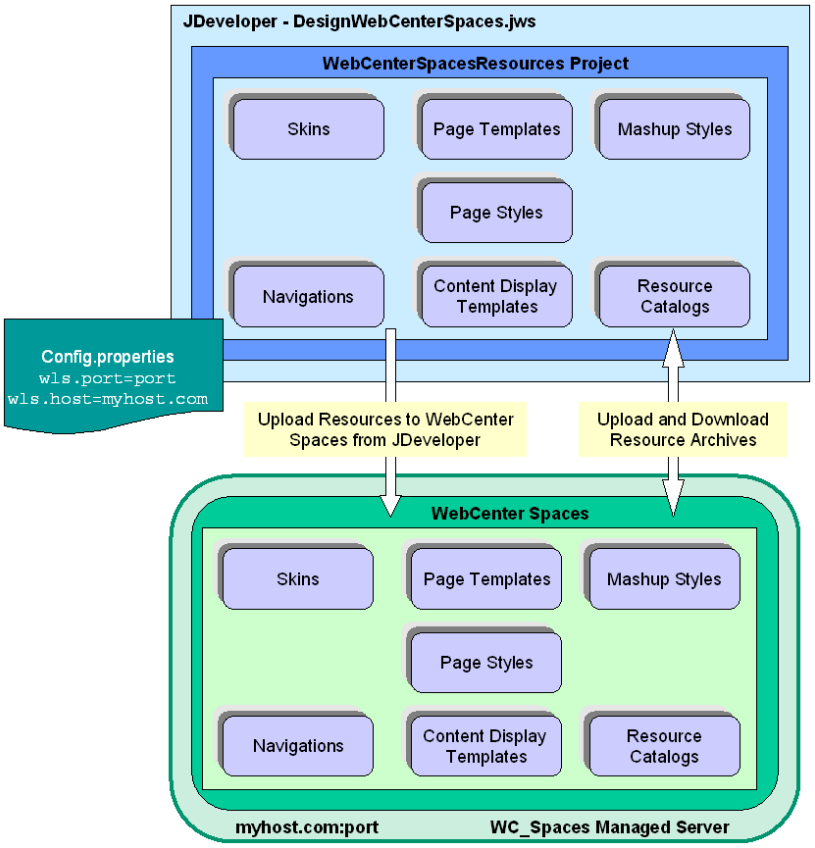
Note: This chapter describes how to use the *WebCenterSpacesResources* project to manage resource customizations. If you need more information on how to build/develop resources, see Section 56.3.4.3, "How to Build Spaces Resources".

When you are ready to deploy new resources, you can upload them directly from JDeveloper to a live Spaces application simply by providing JDeveloper with the name of the host and port where the Spaces application is running.

Note: You do not need to deploy customized resources to a shared library.

Alternatively, you can login to Spaces and upload new resources through the Resource Manager. If you decide to use this method you must save the resource to an export archive (.ear).

Figure 56–9 Designing Resources Using the WebCenterSpacesResources Project



56.3.2 Opening and Exploring the WebCenterSpacesResources Project

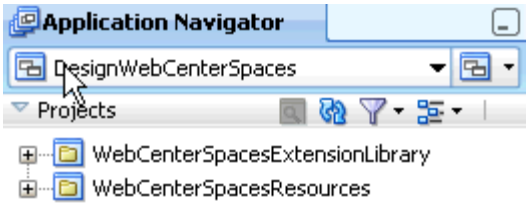
1. Download and unzip the Spaces workspace.

   See Section 56.1, "Downloading a Workspace for Spaces Development".

2. Navigate to the unzip directory, and open `DesignWebCenterSpaces\DesignWebCenterSpaces.jws` in JDeveloper (Figure 56–10).

   This workspace contains a project dedicated to resource development called **WebCenterSpacesResources**.

Figure 56–10 Projects in the DesignWebCenterSpaces Workspace



3. Expand the **WebCenterSpacesResources** project.

   The **WebCenterSpacesResources** project contains several files and folders (Figure 56–11):

Figure 56–11 Exploring the WebCenterSpacesResources Project in JDeveloper

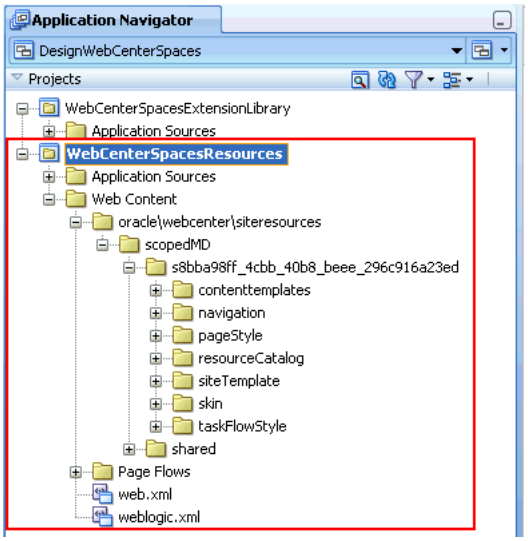


Table 56–3 describes the files and folders in the **WebCenterSpacesResources** project.

Table 56–3 Files and Folders in the WebCenterSpacesResources Project

Configuration Property	Description
\Application Sources	Supporting files and folders.
\Web Content	Contains Spaces resources.

Table 56–3 (Cont.) Files and Folders in the WebCenterSpacesResources Project

Configuration Property	Description
\oracle\webcenter\siteresources	Directory containing Spaces resources. You can develop and manage all your application resources under this folder.
\scopedMD	Contains application-level and Space-level resources. ▪ Application-level resources are stored under the *"default scope"* directory `s8bba98ff_4cbb_40b8_beee_296c916a23ed` Out-of-the box, this directory contains several sample application resources. ▪ Space-level resources are stored under Space-specific directories with the same name as the Space's internal ID. No Space-level resources are provided out-of-the-box.
\<Default Scope ID>	Contains sample application-level resources.
\contenttemplates	Includes a sample Content Presenter display template (`content-default-list-item-view.jsff`) - Sample Content List Item View.
\navigation	includes a sample navigation (`Navigation.xml`) - SampleNavigation.
\pageStyle	Includes a blank page style (`TemplateHome.jspx`) - Sample Home Page Style.
\resourceCatalog	Includes a sample catalog (`ResourceCatalog.xml`) - Sample Home Space Catalog
\siteTemplate	Includes a sample page templates (`Template.jspx`) - Sample Side Navigation **Note:** This directory is the correct directory for developing page templates (even though the directory name is siteTemplate).
\skin	Includes a sample skin (`Skin.css`) - Sample Spaces FX Skin.
\taskFlowStyle	Includes files for a sample mashup style (`pform.jsff` and `pform-definition.xml` - Sample Parameter Form.
\shared	Content directory for images/content used by resources, such as, icons, images, and so on. When you upload a resource to the Spaces application, content in the `\shared` directory is copied to Spaces automatically. Ensure that you copy *all files* that your resources reference to this `\shared` directory and verify that all references to such content point to this location, for example: `<af:image source="/oracle/webcenter/siteresources/scopedMD/shared/mylogo.gif id="pt_12">`
\Page Flows	Supporting files and folders.

56.3.3 Enabling Direct Uploads to the Spaces Application

If you build or modify skins, page templates, and so on, for a Spaces application that is live or running in a test environment you can upload your updates directly to the application from JDeveloper. To enable direct updates from the JDeveloper **WebCenterSpacesResources** project you must:

- Specify the name of the host and port where the Spaces application is running in `config.properties`.
- Check that you have the appropriate roles and permissions to upload the resource to Spaces. You must have the following:
  - **WebLogic Server role** - `monitor` AND
  - **Spaces resource permissions** - For example, to upload a new skin you must have the `Create, Edit, Delete Skins` permission.

Configuring the Spaces hostname and port, enables the *Upload Portal Resource* option in your project. The first time you use this option, within a JDeveloper session, you must enter your Spaces login name and password. For security reasons, your credentials are not saved for future sessions but they are stored in memory for the current JDeveloper session.

Once you are logged in to Spaces you can upload resources, providing, of course, that you have the appropriate permissions to manage Spaces resources.

> **Note:** Out-of-the-box, only the default Fusion Middleware administrator (`weblogic`) has all the required permissions.

To set connection properties in `config.properties` and grant permissions:

1. Open `config.properties`, available at:

   `<Unzip_Directory>\DesignWebCenterSpaces\config.properties`

2. Enter details relating to JDeveloper and your Spaces installation, as shown in Table 56–4:

   The `config.properties` file describes each property and offers examples. The defaults provided are only samples and must be replaced with your own, installation-specific values.

Table 56–4 config.properties Settings to Enable Direct Uploads

Configuration Property	Description
`jdeveloper.install.home.directory`	Base directory where JDeveloper is installed. In addition to `\jdeveloper`, the directory you specify must contain `\wlserver_10.3`, `\modules`, and so on.
`wls.host`	Name of the host machine where the Spaces application is running.
`wls.port`	Port number on which the WLS Administration Console is running.

3. Save your updates to `config.properties`.
4. Verify and, if necessary, request permissions shown in Table 56–5.

Table 56–5 Permissions to Upload and Manage Resources Through JDeveloper

Type	Role or Permissions	Description
WebLogic Server	`monitor`	This role enables you to run the WLST scripts which upload resources from JDeveloper to the Spaces application. See also, "Add users to roles" in *Oracle WebLogic Server Administration Console Help*.
Spaces		
Application-level	`Create, Edit, Delete <resourcetype>`	This permission enables you to create and manage application-level resources for Spaces.
	`Manage Configuration`	This permission gives you access to the Spaces resource administration page. See also, "Managing Application Roles and Permissions" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
Space-level Standard	`Create, Edit, Delete Resources`	These permissions enable you to create and manage resources for a particular Space. Either standard or advanced permissions will apply, depending on the Space.
Space-level Advanced	`Create, Edit, Delete <resourcetype>`	
	`Manage Configuration`	This permission gives you access to the Space's resource administration page. See also, "Managing Roles and Permissions for a Space" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

5. Upload a sample resource, to test the connection, and verify that the resource is available in Spaces:

   For details, see Section 56.3.5, "Uploading Resources Directly to the Spaces Application".

   See also, Section 56.3.7, "Troubleshooting Resource Round-Trip Development"

56.3.4 Editing Spaces Resources in JDeveloper

Oracle recommends that you extend Spaces resources in the JDeveloper project, *WebCenterSpacesResources*. This project provides everything you need to create, modify, and upload the following Spaces resources:

- Page Templates
- Navigations
- Page Styles
- Skins
- Content Display Templates
- Resource Catalogs
- Mashup Styles

Where To Start...

You do not have to develop Spaces resources from scratch. Most developers will download an existing resource from the Spaces application and edit it in JDeveloper. There are two techniques for bringing Spaces resources into JDeveloper. The first is downloading the resource into JDeveloper directly from the server. The second is exporting the resource from Spaces and then importing it into JDeveloper.

For more information, see:

- Downloading a Spaces Resource Directly into JDeveloper
- "Downloading a Spaces Resource" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- Importing a Spaces Resource into JDeveloper

See, also How to Build Spaces Resources.

Application-Level vs Space-Level Resources

The difference between application- and Space-level resources is one of scope. In the Spaces application:

- **Application-level resources are available to all Spaces**.

  You must develop all application-level resources under the WebCenterSpacesResources project directory `oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed`.

  To upload application resources to the Spaces application you must have Spaces permissions: `Application-Manage Configuration` plus `Create, Edit, Delete <Resource_Type>`.

- **Space-level resources are only available within a particular Space**.

  You must develop a resource for a specific Space under the appropriate namespace, so that it can be uploaded to that Space:

  `oracle/webcenter/siteresources/scopedMD/<space_internal_id>`

  Similarly, resources downloaded from an existing Space and imported into JDeveloper, can only be uploaded back to the same Space. You cannot upload such resources to a different Space.

  To upload resources to a particular Space you must have the `Space-Manage Configuration` permission, plus one of:

  `Create, Edit, Delete Resources` (standard permission model)
  `Create, Edit, Delete <Resource_Type>` (advanced permission model)

If you want to upload resources directly from JDeveloper, you must also have the WebLogic Server `monitor` role. See Section 56.3.3, "Enabling Direct Uploads to the Spaces Application".

For information on how to create and edit Spaces resources, see:

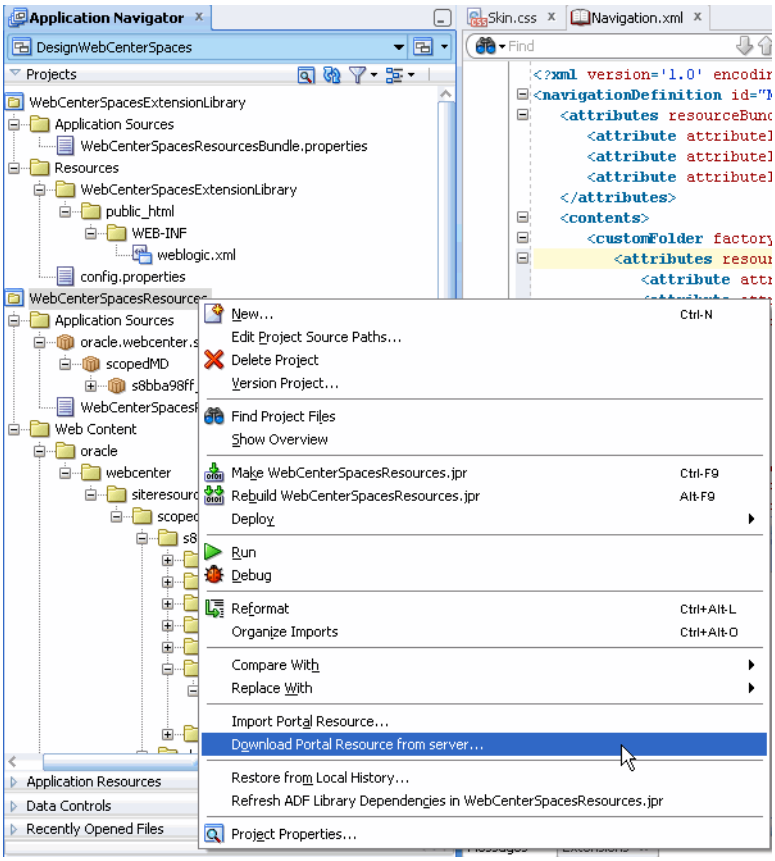
- Section 56.3.4.1, "Downloading a Spaces Resource Directly into JDeveloper"
- Section 56.3.4.2, "Importing a Spaces Resource into JDeveloper"
- Section 56.3.4.3, "How to Build Spaces Resources"

56.3.4.1 Downloading a Spaces Resource Directly into JDeveloper

You can download a Spaces resource directly into JDeveloper from within the WebCenterSpacesResources project.

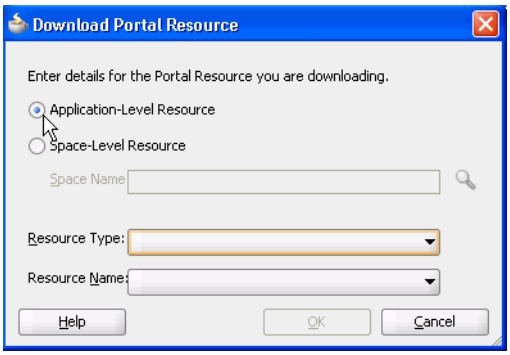
1. Be sure the configuration properties and permissions are set properly, as described in Section 56.3.3, "Enabling Direct Uploads to the Spaces Application." See also Section 56.2.3.2, "Setting Build and Deployment Options (config.properties)."
2. In the Navigator, right-click the WebCenterSpacesResources project and select **Download Portal Resource from Server**, as shown in Figure 56–12.

Figure 56–12 Download Portal Resource from Server Menu



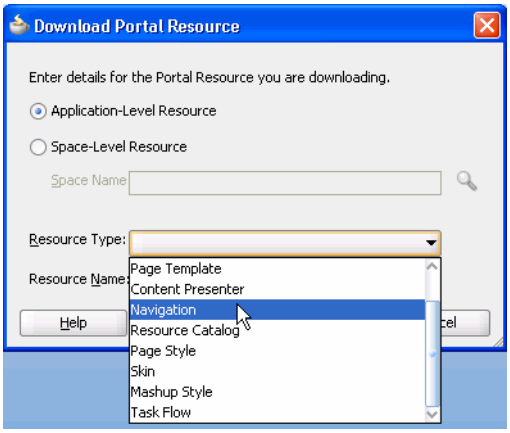
3. In the Download Portal Resources dialog, choose to download either an Application-Level Resource or a Space-Level Resource, as shown in Figure 56–13 See "Application-Level vs Space-Level Resources" for information on these selections.

Figure 56–13 Download Portal Resources Dialog



4. To filter the list of possible resource names, select the type of resource you wish to download from the Resource Type menu, as shown in Figure 56–14.

Figure 56–14 Selecting a Resource Type to Download



5. Finally, select the name of the resource you wish to download from the Resource Name menu.
6. Click **OK**.

56.3.4.2 Importing a Spaces Resource into JDeveloper

When you import a Spaces resource archive (.ear) into the *WebCenterSpacesResources* project the resource is uploaded to the appropriate project directory.

- **Application-level resources are imported to:**

  `WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\`
  `application_ID\resource_type\resource_ID`

  For example, an application-level skin with the resource ID `gsr5a8c2fcc_bc7f_4cba_9254_36df58d66e60` is created under the application directory `s8bba98ff_4cbb_40b8_beee_296c916a23ed`:

  `WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\skin\gsr5a8c2fcc_bc7f_4cba_9254_36df58d66e60`

- **Space-level resources are imported to:**

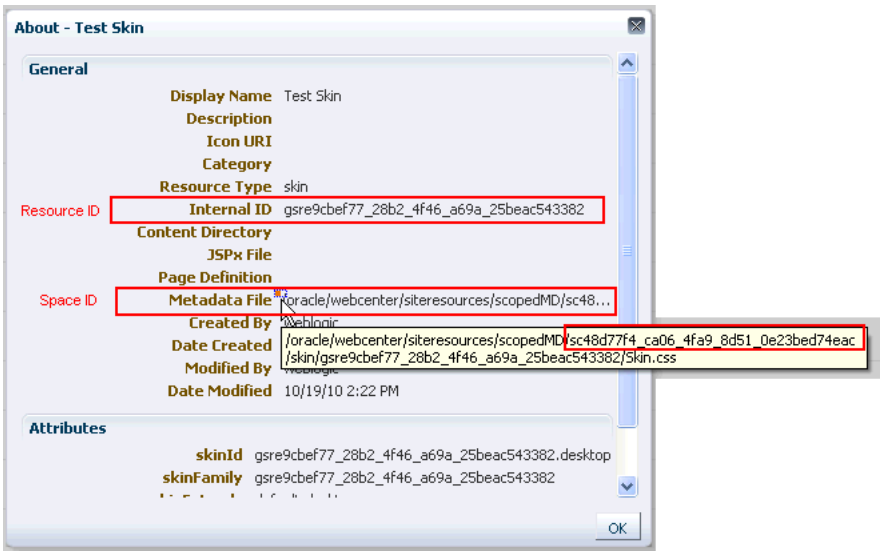
```
WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\
Space_ID\resource_type\resource_ID
```

For example, a Space-level skin with the resource ID `gsre9cbef77_28b2_4f46_a69a_25beac543382` is created under the Space directory `sc48d77f4_ca06_4fa9_8d51_0e23bed74eac`:

```
WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\
sc48d77f4_ca06_4fa9_8d51_0e23bed74eac\skin\gsre9cbef77_28b2_4f46_a69a_25beac543382
```

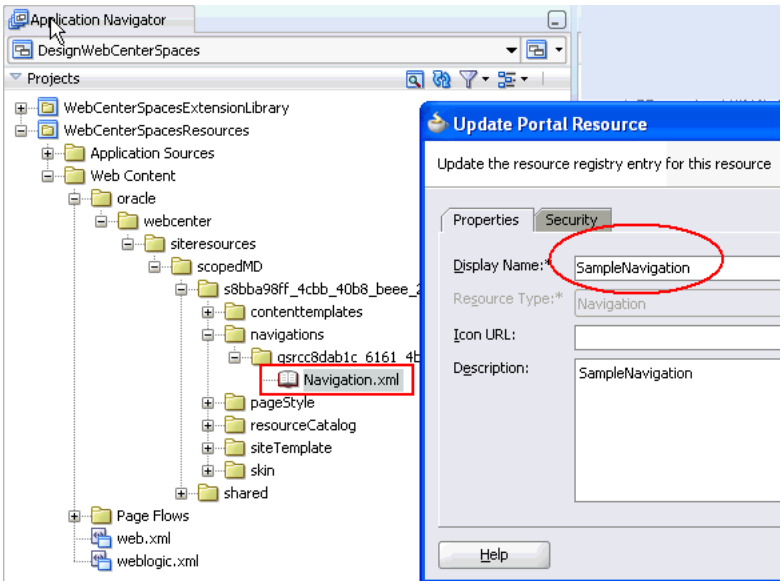
Its important to know the internal IDs of Spaces resources and the parent Space (Space-level resources only) so that you can locate the appropriate resource folders in JDeveloper. In the Spaces application, internal IDs are published in the resource's About dialog (Figure 56–15). For details, see "Viewing Information About a Resource" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Figure 56–15 Checking Resource IDs in Spaces



Internal IDs are not easy to remember like display names. If you want to identify an imported resource you can check its display name using the **Update Resource** option. Navigate to the resource directory, drill down to the resource file (for example, `mynavigation.xml`, `myskin.css`, `mypagetemplate.jpsx`, `mycatalog.xml`), and then click **Update Resource** from the right mouse menu (Figure 56–16).

Figure 56–16 Checking Resource Display Names in JDeveloper



To import a Spaces resource from an archive:

1. In the Application Navigator, right-click the **WebCenterSpacesResources** project, and choose **Import Portal Resource**.
2. In the Import Portal Resource dialog, enter the name or browse for the archive file (.EAR) that contains the resource that you want to import.

   If the archive does not exist yet, login to Spaces and download the resource to an .EAR file. See "Downloading a Spaces Resource" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
3. Click **OK**.
4. Navigate to the resource:
   - Locate application resources under:

     ```
     WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\
     s8bba98ff_4cbb_40b8_beee_296c916a23ed
     ```
   - Locate space-level resources under:

     ```
     WebCenterSpacesResources\oracle\webcenter\siteresources\scopedMD\<Space_GUID>
     ```

56.3.4.3 How to Build Spaces Resources

To learn more about building resources, read the appropriate chapter:

- Page Templates - Working with Page Templates
- Page Styles - Working with Page Styles
- Skins - Creating and Managing Skins
- Navigations - Building a Navigation Model for Your Portal
- Content Display Templates - Creating Content Presenter Display Templates
- Resource Catalogs - Creating and Managing Resource Catalogs

- Mashup Styles - Working with Mashup Styles

56.3.5 Uploading Resources Directly to the Spaces Application

If the Spaces application is up and running you can deploy new or updated resources directly to the application from your JDeveloper environment.

You must have appropriate resource management permissions in the Spaces application to upload resources in this way. If you do not have the correct privileges, you are not allowed to upload resources. See also Section 56.3.3, "Enabling Direct Uploads to the Spaces Application".

1. If you have not done so already, configure Spaces connections details in `config.properties` and request the appropriate resource management permissions.

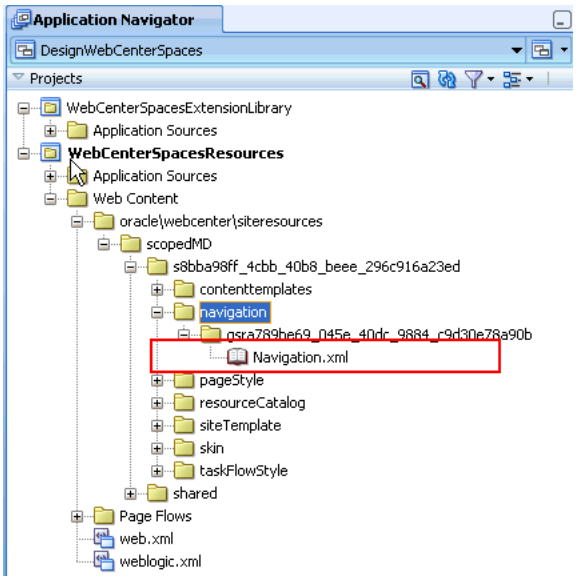
   See also, Section 56.3.3, "Enabling Direct Uploads to the Spaces Application".

2. In JDeveloper, open the **WebCenterSpacesResources** project, and navigate to the resource you want to upload.

   For example, to upload the sample application-level navigation *SampleNavigation*, navigate to (Figure 56–17):

   WebCenterSpacesResources\Web Content\oracle\webcenter\siteresources\scopedMD\s8bba98ff_4cbb_40b8_beee_296c916a23ed\navigation\gsrcc8dab1c_6161_4bb8_8764_127b4ecee01b

Figure 56–17 Navigating to Spaces Resource Folders



3. Verify that the resource is complete.

4. Ensure that all content used by the resource, such as images, icons, and so on, are located under the *content directory* /oracle/webcenter/siteresources/scopedMD/shared and verify that all references to such content point to this location, for example:

```
<af:image
source="/oracle/webcenter/siteresources/scopedMD/shared/mylog
o.gif id="pt_12">
```

5. Right-click the file associated with the resource you want to upload to Spaces, and choose **Upload Portal Resource**.

   There is a single file associated with most resource types. The exception is mashup style which has two files: `definition.xml` and `.jsff`. To upload a mashup style you must right-click the `definition.xml` file.

6. If the selected resource references content under the shared directory, specify whether or not you want to upload the content from JDeveloper.

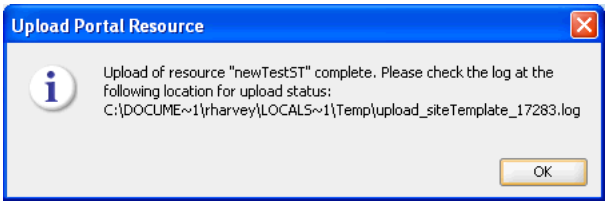
   – Entire content directories are uploaded rather than individual files. For example, if a skin references two files (`...\shared\skins\logos\mylogo.gif` and `...\shared\skins\icons\myicon.gif`), the entire content of both directories (`\logos` and `\icons`) are uploaded.

   – On upload, all files under the resource's content directories are uploaded to the target application and existing files with the same name are overwritten. While you can opt not to upload content directories, it's an all or nothing approach, that is, you cannot only upload files referenced by this resource.

7. If prompted, enter your Spaces username and password and click **OK**.

   An information message displays to indicate that the upload process is complete and also provides the name and location of the upload log file—`upload_<resource_type_id>.log`.

Figure 56–18 Resource Upload Complete Message



8. Review the upload log file.

   The following message indicates that the upload was successful.

   `Imported <temp_log_directory>\<resource_type_id>.ear`

   If you do not see this message, refer to Section 56.3.7, "Troubleshooting Resource Round-Trip Development"

9. Log in to the Spaces application and verify that the resource is available through Resource Manager as expected.

   a. Log in to Spaces.

   b. Navigate to the Resource Manager as follows:

   To verify an application-level resource, navigate to the following URL:

   `http://host:port/webcenter/spaces/admin/resources`

   To verify a Space-level resources, navigate to the following URL:

   `http://host:port/webcenter/spaces/spaceName/admin/resources`

   c. Select the appropriate resource type from the panel on the left, and verify that the resource you just uploaded is available and working as expected.

Dependent images, and so on, must be located in the Spaces content directory. If an image is missing, check that you copied it to the appropriate content directory in JDeveloper (under `/oracle/webcenter/siteresources/scopedMD/shared`).

56.3.6 Exporting Spaces Resources to an Archive

When your resource is ready for use in the Spaces application you can upload it directly to Spaces or you can export the resource to an .EAR file format which you (or someone else) can upload at any time using Spaces Resource Manager.

To export a Spaces resource:

1. In the Application Navigator, navigate to the resource directory and drill down to the resource file (for example, `myskin.css`, `mypagetemplate.jpsx`, `mycatalog.xml`).

   Note: Ensure that all dependent content such as images, icons, style sheets, and so on, are copied to the /oracle/webcenter/siteresources/scopedMD/shared/ directory and verify that all references to such content point to this location, for example: `<af:image source="/oracle/webcenter/siteresources/scopedMD/shared/mylogo.gif id="pt_12">`

2. Click **Export Portal Resourc**e from the right mouse menu.
3. In the Export Portal Resource dialog, enter or browse for the path and file name for the export archive file. For example: `C:\myskin.ear`
4. If the selected resource references content under the shared directory, specify whether or not you want to **Include Content Directories** in the resource archive.

   Do one of the following:

   - Select **Include Content Directories** to export content referenced by the portal resource, such as images and icons, in the export archive.

     Consider the following before exporting content directories for the resource:

     – Entire content directories are exported rather than individual files. For example, if a skin references two files (`...\shared\skins\logos\mylogo.gif` and `...\shared\skins\icons\myicon.gif`), the entire content of both directories (`\logos` and `\icons`) are exported.

     – On import, all files under the content directories are uploaded to the target Spaces application and existing files with the same name are overwritten. While you can opt not to upload content directories on import, it's an all or nothing approach, that is, you cannot upload individual files for this specific resource.

   - Click **No** to exclude content under the `\shared` directory that is referenced by portal resources.

5. Click **OK**.

The steps to upload resource archives to Spaces are available in "Uploading a Spaces Resource" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

56.3.7 Troubleshooting Resource Round-Trip Development

This troubleshooting section includes the following subsections:

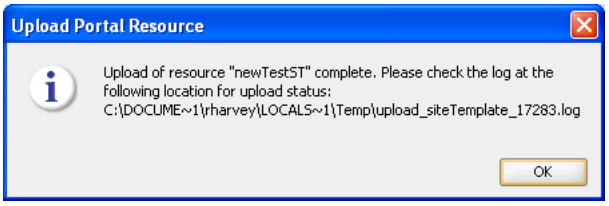
- Examining the Resource Upload Log File
- Unable to Upload Resources to Spaces - Access denied
- Unable to Upload a Resource to Spaces - java.lang.Exception

Examining the Resource Upload Log File

When you upload a resource to the Spaces application progress details are recorded in a log file. The name of the log file includes the type of resource you are uploading and a unique ID, in the format—`upload_`*`resource_type_id`*`.log`. For example, `upload_siteTemplate_17283.log`.

The location of the log file depends on your local `temp` directory settings. The exact log file location displays in an information message such as this:

Figure 56–19 Resource Upload Log File Location



The following log entry indicates that the upload process was successful.

```
Imported <temp_log_directory>\<resource_type_id>.ear
```

Insufficient permissions result in "`Access denied`" or "`java.lang.Exception`" errors, as described below.

Unable to Upload Resources to Spaces - Access denied

The following error displays in the *resource upload log file* if you do not have the `monitor` role:

```
Operation: getScopeName(java.lang.String)  Detail: Access denied. Required roles:
Admin, Operator, Monitor, executing subject: principals=[yourusername]
```

The `monitor` role is a Weblogic Server role. This role enables you to run the WLST scripts which upload resources from JDeveloper to the Spaces application. Ask your Spaces administrator to grant you this role through the WebLogic Server Administration Console. See also, "Add users to roles" in *Oracle WebLogic Server Administration Console Help*.

Unable to Upload a Resource to Spaces - java.lang.Exception

The following error displays in the *resource upload log file* if you do not have permission to manage any resources in Spaces or the particular type of resource you are trying to upload:

```
javax.management.MBeanException at
weblogic.rjvm.ResponseImpl.unmarshalReturn(ResponseImpl.java:234) Root Cause:
Caused by: java.lang.Exception at
oracle.webcenter.portalframework.genericsiteresources.model.lifecycle.SiteResource
sHelper.updateResource(SiteResourcesHelper.java)
```

Ask your Spaces administrator to grant you appropriate permissions through Spaces Administration. Or, if the resource you are trying to upload is for a particular Space, contact the Space moderator to request the required permission. See also, Table 56–5, " Permissions to Upload and Manage Resources Through JDeveloper"".

57

Integrating with Oracle WebCenter Portal: Spaces

This chapter describes how to expose WebCenter Portal: Spaces functionality in a WebCenter Portal: Framework application, using Spaces APIs. Through these APIs, you can create new spaces, manage space membership, retrieve space information, and more, from your Framework applications.

The chapter also describes how to expose information from other applications, such as Framework applications in Spaces.

This chapter includes the following sections:

- Introduction to WebCenter Portal: Spaces
- Exposing Spaces in WebCenter Portal Applications Using APIs
- Using the WebCenter Portal: Spaces REST APIs
- Exposing Enterprise Applications in Spaces

> **Note:** Custom-built task flows that implement Spaces APIs (described in this chapter) can be deployed on WebCenter Portal: Framework applications, but you cannot deploy them to WebCenter Portal: Spaces.

Audience

This section is intended for developers who want to expose Spaces functionality in their WebCenter Portal: Framework applications.

57.1 Introduction to WebCenter Portal: Spaces

WebCenter Portal's Spaces application is a Web-based application, built using the WebCenter Portal: Framework, WebCenter Portal services, and Composer. The Spaces application furnishes all the tools you require to rapidly create portals, communities, and social sites capable of accommodating thousands of users and encompassing diverse populations with differing languages. Figure 57–1 shows a typical Spaces home page.

Figure 57–1 WebCenter Portal: Spaces

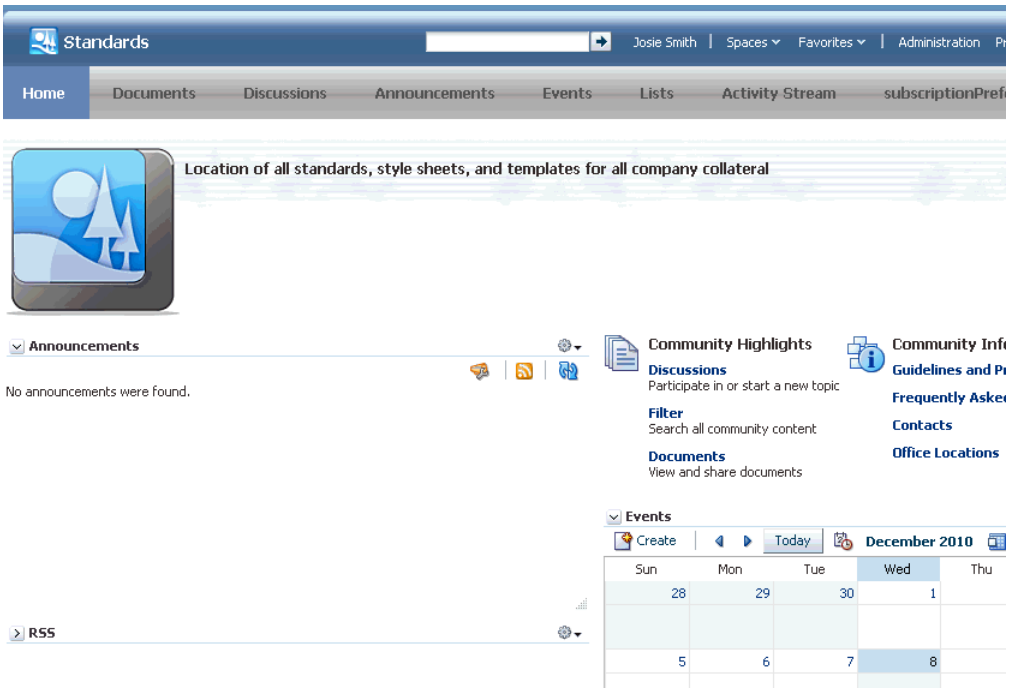


Out of the box, Spaces offers a configurable work environment that enables individuals and groups to work and collaborate more effectively. The application is a self-service solution for managing individual and group interactions. It also provides intuitive tools that allow business users to come together and share information by adding pages and resources such as documents, charts, reports, portlets, business applications, services, and other ADF resources or views.

The Spaces application provides two work environments within a single application: the *Home space* and *group spaces*. The Home space provide each user with a private work area for storing personal content, viewing and responding to business process assignments, and performing many other tasks relevant to his or her unique working day. Group spaces (from hereon referred to as *spaces*) support discrete work areas organized around an area of interest or a common goal.

Spaces support the formation and collaboration of project teams and communities of interest. Spaces bring people together in a virtual environment for ongoing interaction and information sharing—in essence, forming a social network. Figure 57–2 shows an example of a space.

Figure 57–2 Standards - Sample Space



Structurally, spaces comprise pages, many of which are dedicated to a particular service. For example, a *Lists page* provides the means to create and publish multicolumn lists. A *Documents page* provides a central library for uploading, organizing, and managing group content. A *Search page* includes features for saving searches and managing search results. In addition to these and other default pages, a space supports custom pages created by authorized users. Page creation is easy with a wide selection of predefined layouts. With little effort, users can provide pages neatly tailored to the unique needs of their team or community. For more information about spaces, see the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

WebCenter Portal provides various *services* that spaces can use to offer a wide variety of functionality in support of group objectives. For example, a Documents service provides features for uploading and managing content; a Discussions service provides features for creating, managing, and participating in discussion forums. Some other examples of these services include Events, Lists, Announcements, and Search. These services are what helps to make Spaces a productive, collaborative working environment. Services expose subsets of their features and functionality through *task flows*. A task flow is a reusable view that may expose all or a subset of the features available from a particular service. For example, a Recent Documents task flow provides a subset of the functionality offered through the Documents service by listing documents that have recently been opened, added, or affected in some way.

To help users get spaces up and running quickly, they can be based on out-of-the-box templates. Additionally, Spaces allows users to turn any space into a template, so they can design spaces to suit specific audiences and offer them up as templates for others to use. For information about creating templates, see "Creating Custom Space Templates" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Spaces can consume portlets from any WSRP, Oracle PDK-Java, or pagelet producer if the producer is registered first. Therefore, any portlets you create for Framework applications can be used in Spaces too. For more information, see the "Managing

Portlet Producers" chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

57.2 Exposing Spaces in WebCenter Portal Applications Using APIs

This section includes the following subsections:

- Introduction to Spaces APIs
- Case Study 1: Purchasing Application Uses a Space to Evaluate Suppliers
- Case Study 2: Customer Support Center Application Uses a Space to Discuss Customer Escalations
- How to Set Up Your WebCenter Portal Application to Use the Spaces APIs
- How to Provide Spaces Functionality in WebCenter Portal Applications
- How to Handle Exceptions Raised by Spaces APIs
- Finding More Information on Spaces APIs
- Troubleshooting Issues with Spaces APIs

57.2.1 Introduction to Spaces APIs

While using your WebCenter Portal: Framework application, users may encounter situations where Spaces would be useful to help them complete a particular task. In such cases, it would be much less disruptive to remain within the context of the current application, rather than having to switch to the Spaces application. To this end, Spaces provides access to a subset of its functionality through several APIs. Using these APIs, you can integrate powerful Spaces functionality into your Framework application.

You can use Spaces APIs to:

- **Create and manage spaces and space templates.** You can create and delete spaces, and add custom attributes. For more information, see Section 57.2.5.1, "Managing Spaces and Space Template."
- **Manage space membership.** You can add and remove space members. For more information, see Section 57.2.5.2, "Managing Space Membership."
- **Retrieve information about apaces and space templates.** For example, you can retrieve the Spaces application URL or the URL of a specific space. You can also retrieve space and space template metadata. For more information, see Section 57.2.5.3, "Retrieving Information for Spaces and Space Templates."

Spaces APIs are contained within several classes. Table 57–1 lists the different classes and describes the purpose of the APIs within each class.

Table 57–1 Spaces API Classes

Class	Contains APIs for	More Information
`GroupSpaceWSClient`	Creating and managing spaces and space templates Managing space membership Retrieving space information	Section 57.2.5, "How to Provide Spaces Functionality in WebCenter Portal Applications"

Table 57–1 (Cont.) Spaces API Classes

Class	Contains APIs for	More Information
`GroupSpaceWSContext`	Establishing context before calling the APIs	Section 57.2.4.4, "Setting Up the Spaces Client Context"
`GroupSpaceWSException`	Managing exceptions raised by the APIs	Section 57.2.6, "How to Handle Exceptions Raised by Spaces APIs"
`GroupSpaceWSMembers`	Retrieving information about space members	Section 57.2.5.3, "Retrieving Information for Spaces and Space Templates"
`GroupSpaceWSMetadata`	Retrieving space information	Section 57.2.5.3, "Retrieving Information for Spaces and Space Templates"

Spaces can have many different applications. From your perspective, as a WebCenter Portal: Framework application developer, here are a couple of case studies describing scenarios where building and working with Spaces through the APIs might come in useful:

- In a purchasing application, a purchasing manager may create a space to discuss the suitability of a new supplier. For more information, see Section 57.2.2, "Case Study 1: Purchasing Application Uses a Space to Evaluate Suppliers."
- In a Customer Support Center application, a support analyst may create a space to collaborate with other users about a customer escalation. For more information, see Section 57.2.3, "Case Study 2: Customer Support Center Application Uses a Space to Discuss Customer Escalations."

57.2.2 Case Study 1: Purchasing Application Uses a Space to Evaluate Suppliers

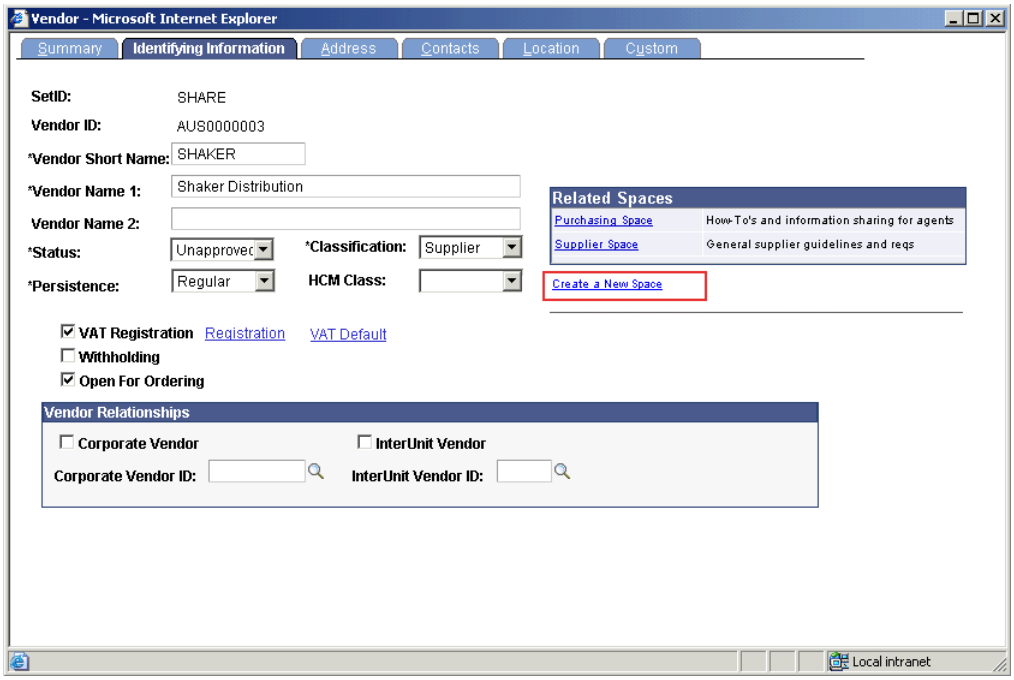
Consider a purchasing application built using the Oracle WebCenter Portal: Framework. This application tracks suppliers, pricing, lead-time requirements, delivery time estimates, and performance history.

Users of the purchasing application must also be able to select and evaluate supplier candidates. Supplier evaluation is a collaborative process; it requires people from various areas of a company. For example, a design engineer and manufacturing representative must verify that an item being purchased meets the required technical specifications; a purchasing agent can negotiate prices, logistics and contractual issues; and a manager or executive has to approve the deal. How could the purchasing application initiate supplier evaluations?

Typically, the purchasing manager receives a purchase requisition from the manufacturing department. Sometimes the purchase order cannot be completed because the requisition cannot be delivered by the usual supplier in the time frame required by manufacturing. Therefore, a new supplier that can meet delivery and pricing requirements must be determined. The purchasing manager can add a candidate supplier into the system but the purchasing manager needs a way to organize and share information, and collaborate with people in and outside his team so that he can assess the supplier.

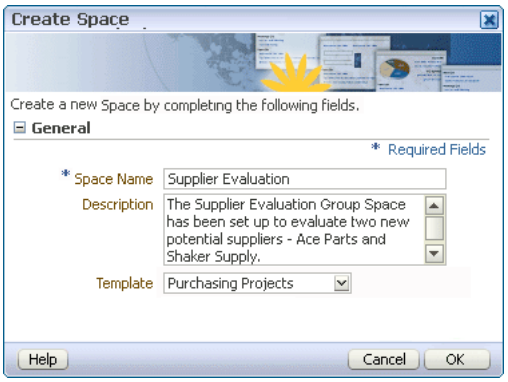
A space can provide this collaborative environment. So the purchasing application includes a call to the `createGroupSpace` API, and this enables the purchasing manager to click a link (**Create a New Space**) that displays a "Create Space" dialog, directly from the purchasing application, as shown in Figure 57–3.

Figure 57–3 Purchasing Application Includes a Space Creation Link



When the purchasing manager clicks the link a custom dialog prompts him for some information. The purchasing manager enters a name and description for the space, and also selects a template (Purchasing Projects) as shown in Figure 57–4. The Purchasing Projects template sets up the space quickly so it is ready to use right away. For example, the template defines which services are required (events, a document library, and so on) and any custom pages that may be required. Because the APIs enable you to create your own "Create Space" dialog, you can apply your own look and feel and terminology.

Figure 57–4 Custom Create Space Dialog



In this scenario, the purchasing manager chooses the Purchasing Projects template from the template list. Another approach would be to have the purchasing application pass in a default template value. With this additional default, there would be one less thing (determining which template to use) for the purchasing manager to think about. You could even generate the name of the space from the Supplier ID so that the purchasing manager would not have to enter any details at all. The space could then be created with just the click of a link.

When the purchasing manager clicks **OK**, the new Supplier Evaluation space displays, similar to that shown in Figure 57–5.

Figure 57–5 Supplier Evaluation Space

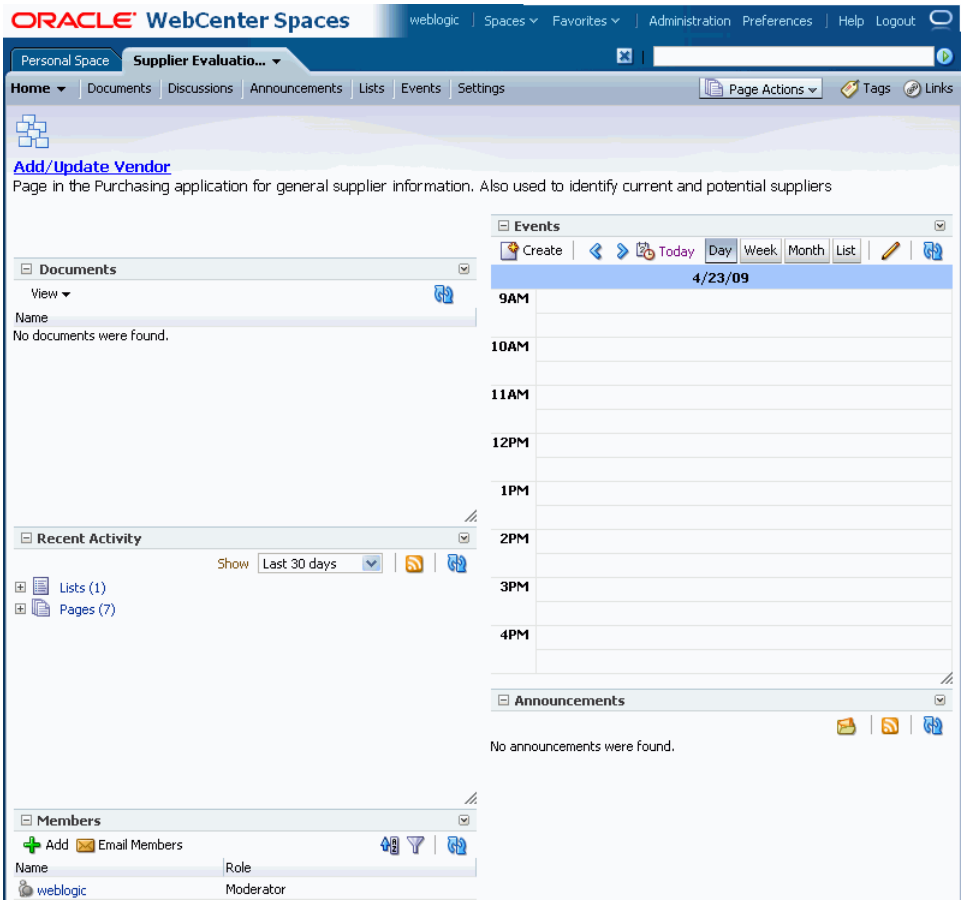


Figure 57–5 shows a Home page for the Supplier Evaluation space, which includes an events calendar, documents the group may find useful, an area for announcements, and so on. Each of these areas was determined by the Purchasing Projects template. In addition a link to the purchasing application transaction instance from which the space was created is also provided. Clicking this link (**Add/Update Vendors**) displays the screen Add/Update Vendor transaction for the supplier Shaker Distribution.

In the Spaces application, the purchasing manager becomes the moderator of the space automatically. The moderator can add content to the space, initiate discussions, invite members, and collaborate with interested parties. Once consensus is reached regarding the supplier, the purchasing manager is can approve or reject the supplier concerned.

57.2.3 Case Study 2: Customer Support Center Application Uses a Space to Discuss Customer Escalations

Consider a Customer Support Center application built using the Oracle WebCenter Portal: Framework that tracks customer calls and issues.

A support analyst is notified that a customer has escalated the service request that the analyst has been working on. The analyst knows that she can find a quicker resolution to the issue if she can involve other people from different areas of the company. For example:

- The project manager whose team implemented the project can provide more in depth knowledge of the project.
- An account manager who has been in constant touch with the customer can provide specific information about the implementation of the project at the customer site.
- Another support analyst, who has worked on a similar escalation before, can provide information about how that escalation was resolved.

This problem can be solved collaboratively using a space. The support analyst creates a space from within the Customer Support Center application and adds the required members. These members are then notified and use the space to start discussing the customer situation. The support analyst views their updates to the space inside the support application, navigating to the space whenever necessary to obtain more specific details. Based on the information she gets from the other members of the space, she can diagnose the problem and offer the customer a solution very quickly.

57.2.4 How to Set Up Your WebCenter Portal Application to Use the Spaces APIs

Before you can use the Spaces APIs you must ensure that the Spaces application is up and running and that your project is set up correctly in JDeveloper.

This section includes the following subsections:

- Verifying That the Spaces Application Is Up and Running
- Setting Up the WebCenter Portal Application to Use Spaces APIs
- Securing the Connection Between the Application and Spaces
- Setting Up the Spaces Client Context

57.2.4.1 Verifying That the Spaces Application Is Up and Running

If you want to use the SpacesAPIs, the Spaces application must be up and running.

To verify that Spaces is up and running:

1. Start Spaces. The URL is:

   `http://host:port/webcenter`

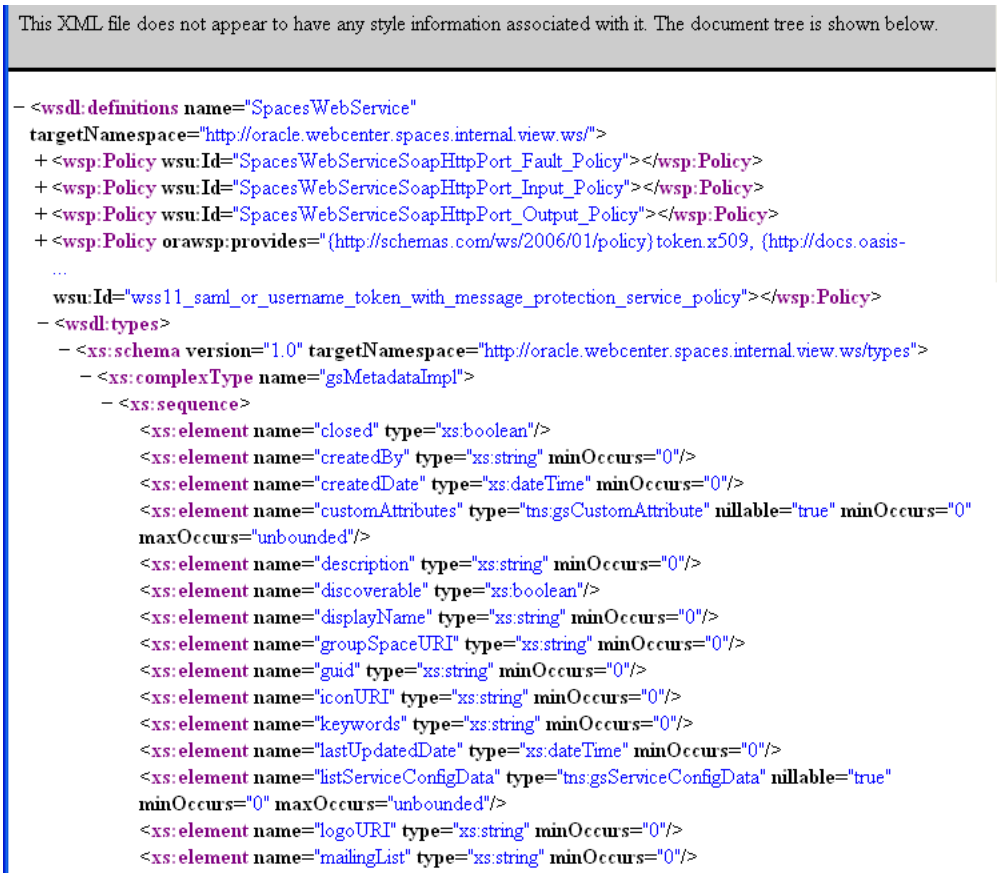
   You do not need to log in.

2. Verify that the Spaces Web service is up and running. The URL is:

   `http://host:port/webcenter/SpacesWebService?WSDL`

   You should see a page similar to the one shown in Figure 57–6. If you do not see this page, contact your Fusion Middleware administrator.

Figure 57–6 SpacesWebService WSDL



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <wsdl:definitions name="SpacesWebService"
  targetNamespace="http://oracle.webcenter.spaces.internal.view.ws/">
  + <wsp:Policy wsu:Id="SpacesWebServiceSoapHttpPort_Fault_Policy"></wsp:Policy>
  + <wsp:Policy wsu:Id="SpacesWebServiceSoapHttpPort_Input_Policy"></wsp:Policy>
  + <wsp:Policy wsu:Id="SpacesWebServiceSoapHttpPort_Output_Policy"></wsp:Policy>
  + <wsp:Policy orawsp:provides="{http://schemas.com/ws/2006/01/policy}token.x509, {http://docs.oasis-
    ...
    wsu:Id="wss11_saml_or_username_token_with_message_protection_service_policy"></wsp:Policy>
  - <wsdl:types>
    - <xs:schema version="1.0" targetNamespace="http://oracle.webcenter.spaces.internal.view.ws/types">
      - <xs:complexType name="gsMetadataImpl">
        - <xs:sequence>
            <xs:element name="closed" type="xs:boolean"/>
            <xs:element name="createdBy" type="xs:string" minOccurs="0"/>
            <xs:element name="createdDate" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="customAttributes" type="tns:gsCustomAttribute" nillable="true" minOccurs="0"
            maxOccurs="unbounded"/>
            <xs:element name="description" type="xs:string" minOccurs="0"/>
            <xs:element name="discoverable" type="xs:boolean"/>
            <xs:element name="displayName" type="xs:string" minOccurs="0"/>
            <xs:element name="groupSpaceURI" type="xs:string" minOccurs="0"/>
            <xs:element name="guid" type="xs:string" minOccurs="0"/>
            <xs:element name="iconURI" type="xs:string" minOccurs="0"/>
            <xs:element name="keywords" type="xs:string" minOccurs="0"/>
            <xs:element name="lastUpdatedDate" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="listServiceConfigData" type="tns:gsServiceConfigData" nillable="true"
            minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="logoURI" type="xs:string" minOccurs="0"/>
            <xs:element name="mailingList" type="xs:string" minOccurs="0"/>
```

For more information about setting up Spaces, see the "Getting the Spaces application Up and Running" chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

57.2.4.2 Setting Up the WebCenter Portal Application to Use Spaces APIs

Before you can call the Spaces APIs in your WebCenter Portal: Framework application, you must ensure that your application contains the correct libraries and that there is a connection to the Spaces Web service.

To set up your application to use the Spaces APIs:

1. Start JDeveloper.
2. In the Application Navigator, expand the application for which you want to provide Spaces functionality.

   The application should be based on the WebCenter Portal Application template.
3. Right-click the view-controller project and choose **Project Properties.**
4. Select **Libraries and Classpath.**
5. Check that the following libraries are available in your project, adding any as necessary:
   - ADF DVT Faces Runtime
   - ADF Model Generic Runtime

- ADF Model Runtime
- JAX-RPC Client
- JAX-WS Client
- JAXB
- Oracle JEWT
- Spaces Client
- *JDEV_HOME*/jdeveloper/oracle_common/modules/oracle.adf.model_11.1.1/adfm.jar
- *JDEV_HOME*/jdeveloper/oracle_common/modules/oracle.xdk_11.1.1/xml.jar

6. Click **OK.**
7. In the Application Resources panel of the Application Navigator, right-click **Connections** and choose **New Connection > URL.**
8. In the **Name** field, enter `SpacesWebServiceEndpoint`.
9. In the **URL Endpoint** field, enter the Spaces Web service URL:

   `http://host:port/webcenter/SpacesWebService`

10. Click **OK.**

    The connection information is added to the `connections.xml` file in the application's `META-INF` directory, for example:

    `C:\JDeveloper\mywork\mySpacesApplication\.adf\META-INF`

    If the `connections.xml` file does not exist, it is created.

11. Open the `connections.xml` file and verify that the code shown in Example 57–1 appears:

Example 57–1 Spaces Web Service Endpoint URL Connection

```
<Reference name="SpacesWebServiceEndpoint"
className="oracle.adf.model.connection.url.HttpURLConnection">
<Factory className="oracle.adf.model.connection.url.URLConnectionFactory"/>
<RefAddresses>
  <XmlRefAddr addrType="SpacesWebServiceEndpoint">
    <Contents>
      <urlconnection name="SpacesWebServiceEndpoint"
      url="http://host:port/webcenter/SpacesWebService"/>
    </Contents>
  </XmlRefAddr>
</RefAddresses>
</Reference>
```

If you need to set the SpacesWeb service endpoint at runtime, you can use the `setGroupSpaceWebServiceEndpoint` API (Example 57–2). You can write a wrapper API that takes the endpoint as a parameter and then calls `setGroupSpaceWebServiceEndpoint`, passing the parameter. An exception is reported as INFO in the log file if the endpoint is not set in `connections.xml`.

Example 57–2 Setting the Spaces Web Service Endpoint at Runtime

```
client.setGroupSpacesWebServiceEndpoint("http://myserver.example.com/webcenter/Spa
```

```
cesWebService");
```

57.2.4.3 Securing the Connection Between the Application and Spaces

Before using the Spaces APIs in your WebCenter Portal: Framework application, you must ensure that the communication between the application (the consumer) and Spaces (the producer) is secure. This is so that the identity of the user invoking the APIs is propagated to Spaces in a secure manner where the integrity and confidentiality of the communication is maintained.

To do this, the Spaces APIs use a policy of SAML based token passing with message protection. Your administrator must create a Java keystore and update the credential store so that Spaces can verify the authenticity of the security tokens received from your application. You must then register this keystore and update the credential store using JDeveloper.

For information about the steps that the administrator must perform, see "Securing Spaces for Applications Consuming Spaces Client APIs with WS-Security" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

57.2.4.4 Setting Up the Spaces Client Context

Before calling any of the Spaces APIs in your application code, a few setup steps are required.

To set up the Spaces client context

1. Domain-wide configuration settings are used when you call Spaces APIs from an application deployed on WebLogic Server (WLS), unless you specifically choose to override them. To override or set security configuration parameters, use the APIs provided by the `GroupSpaceWSContext` class.

   In particular, you must set the SAML issuer name and the public key alias for Spaces, which are required for data encryption. If the Spaces Web service endpoint is not set in the `connections.xml` file, you can set this too.

   For example:

   ```
   GroupSpaceWSContext context = new GroupSpaceWSContext();

   context.setEndPoint("endPointUrl");
   context.setSamlIssuerName("samlIssuer");
   context.setRecipientKeyAlias("producer");

   groupSpaceWSClient = new GroupSpaceWSClient(context);
   ```

   Where:

   - *`endPointUrl`* is the Spaces Web service endpoint, for example, `http://server.example.com:8912/webcenter/SpacesWebService`.
   - *`samlIssuer`* is the issuer URI of the SAML Authority issuing assertions for this SAML Asserting Party, for example, `http://www.oracle.com`. Out-of-the-box, `http://www.oracle.com` is the only trusted SAML issuer.
   - *`producer`* is the public key alias for Spaces, for example, `orakey` if you are using a simple topology.

     For information about the steps that the administrator must perform, see "Securing Spaces for Applications Consuming Spaces Client APIs with

WS-Security" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

For a full list of the APIs provided by the `GroupSpaceWSContext` class, see the *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

2. Initialize the Spaces client by passing the context. For example:

```
GroupSpaceWSClient client = new GroupSpaceWSClient(context);
```

3. Once you have initialized the client, check everything is set up correctly by calling:

```
getWebCenterSpacesURL();
```

If the correct URL is returned, everything is set up correctly and you can start using the Spaces APIs.

57.2.5 How to Provide Spaces Functionality in WebCenter Portal Applications

The Spaces APIs enable you to perform common space-related operations within a WebCenter Portal: Framework application. Most of these APIs are provided by the `GroupSpaceWSClient` class. The APIs can be grouped into three main categories:

- Managing Spaces and Space Template
- Managing Space Membership
- Retrieving Information for Spaces and Space Templates

Table 57–2 lists the APIs in the `GroupSpaceWSClient` class.

Table 57–2 APIs for Performing Common Spaces Operations

Spaces API	Category	Description
`createGroupSpace`	Managing Spaces and Space Templates	Creates a new space based on a template. See Section 57.2.5.1.1, "Creating a Space."
`setCustomAttribute`	Managing Spaces and Space Templates	Creates one or more custom attributes for a space. See Section 57.2.5.1.2, "Creating a Custom Attribute."
`deleteGroupSpace`	Managing Spaces and Space Templates	Permanently removes a space from Spaces. See Section 57.2.5.1.4, "Deleting a Space."
`createGroupSpaceTemplate`	Managing Spaces and Space Templates	Creates a new space template based on an existing space. See Section 57.2.5.1.3, "Creating a Space Template."
`addMember`	Managing Space Membership	Makes a user (or a group) a space member with a specific role. See Section 57.2.5.2.1, "Adding Members to a Space."
`inviteMember`	Managing Space Membership	Invites a list of users to become members of a space. See Section 57.2.5.2.2, "Inviting Users to Join a Space."
`removeMember`	Managing Space Membership	Revokes space membership. See Section 57.2.5.2.3, "Removing Members from a Space."

Table 57–2 (Cont.) APIs for Performing Common Spaces Operations

Spaces API	Category	Description
`getRoles`	Managing Space Membership	Retrieves all the roles for a given space. See Section 57.2.5.2.4, "Retrieving Role Information."
`getGroupSpaces`	Retrieving Information for Spaces and Space Templates	Retrieves a list of spaces for a given query string. See Section 57.2.5.3.1, "Retrieving a List of Spaces."
`getPublicGroupSpaces`	Retrieving Information for Spaces and Space Templates	Retrieves a list of public spaces for a given query string. See Section 57.2.5.3.2, "Retrieving a List of Public Spaces."
`getDiscoverableGroupSpaces`	Retrieving Information for Spaces and Space Templates	Retrieves a list of discoverable spaces for a given query string. See Section 57.2.5.3.3, "Retrieving a List of Discoverable Spaces."
`getGroupSpaceMetadata`	Retrieving Information for Spaces and Space Templates	Retrieves information (metadata) about the space. See Section 57.2.5.3.4, "Retrieving Space Metadata."
`getGroupSpaceMetadataByGuid`	Retrieving Information for Spaces and Space Templates	Retrieves information (metadata) about a space given the space's GUID. See Section 57.2.5.3.4, "Retrieving Space Metadata."
`getGroupSpaceTemplates`	Retrieving Information for Spaces and Space Templates	Retrieves a list of space templates for a given query string. See Section 57.2.5.3.5, "Retrieving a List of Space Templates."
`getGroupSpaceTemplateMetadata`	Retrieving Information for Spaces and Space Templates	Retrieves information (metadata) about the space template. See Section 57.2.5.3.6, "Retrieving Space Template Metadata."
`getGroupSpaceTemplateMetadataByGuid`	Retrieving Information for Spaces and Space Templates	Retrieves information (metadata) about the space template given the template's GUID. See Section 57.2.5.3.6, "Retrieving Space Template Metadata."
`getWebCenterSpacesURL`	Retrieving Information for Spaces and Space Templates	Retrieves the Spaces application URL. See Section 57.2.5.3.7, "Retrieving the Spaces URL."
`getGroupSpaceURL`	Retrieving Information for Spaces and Space Templates	Retrieves a space URL. See Section 57.2.5.3.8, "Retrieving a Space URL."
`getServiceRSSFeedURL`	Retrieving Information for Spaces and Space Templates	Retrieves RSS feed URLs for the specified space service.
`getServiceRSSFeedURLbyGuid`	Retrieving Information for Spaces and Space Templates	Retrieves RSS feed URL for a given space by GUID and a given service.

57.2.5.1 Managing Spaces and Space Template

Use the following Spaces APIs to manage your spaces and space templates:

- `createGroupSpace`
- `setCustomAttribute`
- `createGroupSpaceTemplate`

- `deleteGroupSpace`

Before you begin

Before using any of these APIs, you must complete all the steps listed in Section 57.2.4, "How to Set Up Your WebCenter Portal Application to Use the Spaces APIs."

This section includes the following subsections

- Creating a Space
- Creating a Custom Attribute
- Creating a Space Template
- Deleting a Space

57.2.5.1.1 Creating a Space Use the `createGroupSpace` API to create a space that is based on an existing space template.

To use this API, specify:

- The internal name of the new space. This name can contain blank spaces.
- A display name for the new space.
- A description of the space. Although this is not mandatory, we recommend that you provide a description to help users identify the space's purpose.
- The name of the space template you want to use. This should be the internal name of the template, not the display name.

Optionally, you can provide a comma separated list of keywords to help users locate the space in searches.

Example 57–3 creates a space with the name *Databases* that is based on the *CommunityofInterest* template. The example also specifies two search keywords (*databases* and *Oracle*).

Example 57–3 Creating a Space

```
//create the space
GroupSpaceWSMetadata gsMetadata =
client.createGroupSpace("Databases", "Databases" "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");
//print the space GUID to provide confirmation of creation
System.out.println("GUID: " + gsMetadata.getGuid());
```

For an example of how to provide custom attributes for a new space, see Section 57.2.5.1.2, "Creating a Custom Attribute."

57.2.5.1.2 Creating a Custom Attribute Every space comes with built-in attributes such as name, description, date created, icon, and so on. In addition, you can define custom attributes that store additional information (metadata) that is unique to the space and its characteristics.

Use the `setCustomAttribute` API to specify a custom attribute for a space. To use this API, specify the name of the space and a name, description, type, and value for the custom attribute.

Example 57–4 creates the Databases space and then creates a custom attribute for that space. The attribute is named *Vendors,* with the description *List of vendors.* It takes string values of *Oracle* and *IBM.*

Example 57–4 Creating a Custom Attribute

```
//create the space
client.createGroupSpace("Databases", "Databases", "A community for people
interested in databases", null, "CommunityofInterest");
//create the custom attribute
client.setCustomAttribute("Databases", "Vendors", "List of vendors",
"java.lang.String", "Oracle, IBM");
```

57.2.5.1.3 Creating a Space Template Use the `createGroupSpaceTemplate` API to create a space template based on an existing space. To use this API specify a name, display name, and description for the template, and the name of the space to use to create the template.

Example 57–5 creates a space template based on the *Databases* space.

Example 57–5 Creating a Space Template

```
GroupSpaceWSMetadata templMetadata =
client.createGroupSpaceTemplate("DatabasesTemplate", "Databases Template",
"A template based on the Databases space", "Databases");
//print the template GUID to provide confirmation of creation
System.out.println("GUID: " + templMetadata.getGuid());
```

57.2.5.1.4 Deleting a Space Use the `deleteGroupSpace` API to permanently delete a space from the Spaces application. To use this API, specify the name of the space to delete. The API returns a boolean value to indicate whether the deletion was successful.

Example 57–6 deletes the space with the name *Databases*. The example uses the boolean value returned by the API to print a message about the success or failure of the operation.

Example 57–6 Deleting a Space

```
//delete the space
Boolean success = client.deleteGroupSpace("Databases");
//print a message depending on the result of the deletion
if(success)
{
  System.out.println("Operation Succeeded");
}
else
{
  System.out.println(Operation Failed");
}
```

57.2.5.2 Managing Space Membership

Use the following Spaces APIs to manage space membership:

- `addMember`
- `inviteMember`
- `removeMember`
- `getRoles`

Before you begin

Before using any of these APIs, you must complete all the steps listed in Section 57.2.4, "How to Set Up Your WebCenter Portal Application to Use the Spaces APIs."

This section includes the following subsections:

- Adding Members to a Space
- Inviting Users to Join a Space
- Removing Members from a Space
- Retrieving Role Information

57.2.5.2.1 Adding Members to a Space

Use the `addMember` API to give users (and groups) space membership and assign the new members to a specific role. To use this API, specify the name of the space, and a list of users/groups. The list must contain objects of type `GroupSpaceWSMembers` that specify the user/group name and the role to give that user/group in the space.

You must specify the name of a valid user or user group that exists in the Spaces identity store. For the role, choose one of the default roles (Moderator, Participant, Viewer) or a custom role (if any). To retrieve a list of the available roles for a space, use the `getRoles` API. For more information, see Section 57.2.5.2.4, "Retrieving Role Information." For more information about roles, see the "Managing Roles and Permissions for a Space" section in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Example 57–7 makes users (of type `GroupSpaceWSMembers`) named *Pat* and *Vicki* and everyone in the *Sales* and *Marketing* user group, members of the *Databases* space. *Pat* and everyone in *Sales* and *Marketing* are granted the *Viewer* role. *Vicki* is assigned the *Participant* role.

> **Note:** Use `setGroup(true)` to indicate when you are adding a *group of users*.

Example 57–7 Adding Members to a Space

```
//create the list of users
List addMem = new ArrayList();
//create the GroupSpaceWSMembers objects
GroupSpaceWSMembers mem1 = new GroupSpaceWSMembers("pat", "Viewer");
GroupSpaceWSMembers mem2 = new GroupSpaceWSMembers("vicki", "Participant");
GroupSpaceWSMembers grp1 = new GroupSpaceWSMembers("Sales", "Viewer");
grp1.setGroup(true);
GroupSpaceWSMembers grp2 = new GroupSpaceWSMembers("Marketing", Viewer");
grp2.setGroup(true);
//add the GroupSpaceWSMembers objects to the list of users
addMem.add(mem1);
addMem.add(mem2);
addMem.add(grp1);
addMem.add(grp2);
//add the users to the Space
client.addMember("Databases", addMem);
//print a list of members to confirm the new members were added
GroupSpaceWSMetadata memMetData = client.getGroupSpaceMetadata("Databases");
for (GroupSpaceWSMembers mems: memMetData.getMembers())
{
  System.out.println(mems.getMember());
  System.out.println(mems.getRole());
```

```
}
```

57.2.5.2.2 Inviting Users to Join a Space

Use the `inviteMember` API to invite users to become members of a space. To use this API, specify the name of the space and a list of users to invite. The list must contain objects of type `GroupSpaceWSMembers` that specify the user name and the role to give that user in the space. An invitation to join the space is sent to each user, and each user can then accept or reject that invitation.

You must specify a valid user name that exists in the Spaces identity store. For the role, choose one of the default roles (Moderator, Participant, Viewer) or a custom role (if any). To retrieve a list of the available roles for a space, use the `getRoles` API. For more information, see Section 57.2.5.2.4, "Retrieving Role Information." For more information about roles, see the "Managing Roles and Permissions for a Space" section in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Example 57–8 invites users (of type `GroupSpaceWSMembers`) named *Pat* and *Vicki* to become members of the *Databases* space with *Viewer* and *Participant* roles respectively.

Example 57–8 Inviting Users to Join a Space

```
//create the list of users
List inviteMem = new ArrayList();
//create the GroupSpaceWSMembers objects
GroupSpaceWSMembers mem1 = new GroupSpaceWSMembers("pat", "Viewer");
GroupSpaceWSMembers mem2 = new GroupSpaceWSMembers("vicki", "Participant");
//add the GroupSpaceWSMembers objects to the list of users
inviteMem.add(mem1);
inviteMem.add(mem2);
//invite the list of users to join the space
client.inviteMember("Databases", inviteMem);
```

57.2.5.2.3 Removing Members from a Space

Use the `removeMember` API to revoke space membership. To use this API, specify the name of the space, and a list of users. The list must contain objects of type `GroupSpaceWSMembers` that specify the user name. To obtain a list of current space members, use the `getGroupSpaceMetadata` API. For more information, see Section 57.2.5.3.4, "Retrieving Space Metadata."

Example 57–9 removes users (of type `GroupSpaceWSMembers`) *Pat* and *Vicki* from the membership of the *Databases* space.

Example 57–9 Removing Members from a Space

```
//create the list of users
List remMem = new ArrayList();
//create the GroupSpaceWSMembers objects
GroupSpaceWSMembers mem1 = new GroupSpaceWSMembers("pat");
GroupSpaceWSMembers mem2 = new GroupSpaceWSMembers("vicki");
2 lines
//add the GroupSpaceWSMembers objects to the list of users
remMem.add(mem1);
remMem.add(mem2);

//remove the users from the space
client.removeMember("Databases", remMem);
//print a list of members to confirm the members were removed
GroupSpaceWSMetadata memMetData = client.getGroupSpaceMetadata("Databases");
for (GroupSpaceWSMembers mems: memMetData.getMembers())
{
```

```
  System.out.println(mems.getMember());
  System.out.println(mems.getRole());
}
```

57.2.5.2.4 Retrieving Role Information Use the `getRoles` API to retrieve all the roles for a given space. This is useful for determining which roles are available when adding or inviting members to a space. Roles include out-of-the box roles (Moderator, Participant, Viewer) and also custom roles (if any). To use this API, specify the name of the space.

Example 57–10 retrieves and displays role information for the *Databases* space.

Example 57–10 Retrieving Role Information

```
//retrieve the list of roles
List<String> roles = client.getRoles("Databases");
//print the list of roles
for (String role: roles)
{
  System.out.println(role);
}
```

57.2.5.3 Retrieving Information for Spaces and Space Templates

Use the following Spaces APIs to retrieve space information:

- `getGroupSpaces`
- `getPublicGroupSpaces`
- `getDiscoverableGroupSpaces`
- `getGroupSpaceMetadata`
- `getGroupSpaceMetadataByGuid`
- `getGroupSpaceTemplates`
- `getGroupSpaceTemplateMetadata`
- `getGroupSpaceTemplateMetadataByGuid`
- `getWebCenterSpacesURL`
- `getGroupSpaceURL`

Before you begin

Before using any of these APIs, you must complete all the steps listed in Section 57.2.4, "How to Set Up Your WebCenter Portal Application to Use the Spaces APIs."

This section includes the following subsections

- Retrieving a List of Spaces
- Retrieving a List of Public Spaces
- Retrieving a List of Discoverable Spaces
- Retrieving Space Metadata
- Retrieving a List of Space Templates
- Retrieving Space Template Metadata

- Retrieving the Spaces URL
- Retrieving a Space URL
- Retrieving RSS Feed URLs for Space Services

57.2.5.3.1 Retrieving a List of Spaces Use the `getGroupSpaces` API to obtain a list of spaces that match a given query string. To use this API, specify a query string. A null value query string returns a list of *all* spaces that are accessible to the current user.

The API returns spaces that are accessible to the current user, up to a maximum of 500.

Example 57–11 returns a list of spaces containing the string *Database.*

Example 57–11 Retrieving a List of Specific Spaces

```
List<String> allGroupSpaces = client.getGroupSpaces("Database");
```

Example 57–12 returns a list of *all* spaces to which the current user has access. This is achieved by specifying a *null* query string.

Example 57–12 Retrieving a List of All Spaces

```
List<String> allGroupSpaces = client.getGroupSpaces(null)
```

57.2.5.3.2 Retrieving a List of Public Spaces Use the `getPublicGroupSpaces` API to obtain a list of public spaces that match a given query string. To use this API, specify a query string. A null value query string returns a list of *all* public spaces.

The API returns spaces that are accessible to all users, even those who are not logged in to the Spaces application, up to a maximum of 500.

Example 57–13 returns a list of public spaces containing the string *Database*.

Example 57–13 Retrieving a List of Specific Public Spaces

```
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces("Database");
```

Example 57–14 returns a list of *all* public spaces. This is achieved by specifying a *null* query string.

Example 57–14 Retrieving a List of All Public Spaces

```
List<String> allPublicGroupSpaces = client.getPublicGroupSpaces(null)
```

57.2.5.3.3 Retrieving a List of Discoverable Spaces Use the `getDiscoverableGroupSpaces` API to obtain a list of discoverable spaces that match a given query string. To use this API, specify a query string. A null value query string returns a list of *all* discoverable spaces.

The API returns spaces that are accessible to all users who are logged in to the Spaces application, up to a maximum of 500.

Example 57–13 returns a list of discoverable spaces containing the string *Database*.

Example 57–15 Retrieving a List of Discoverable spaces

```
List<String> allDiscoverableGroupspaces =
client.getDiscoverableGroupspaces("Database");
```

Example 57–14 returns a list of *all* discoverable spaces. This is achieved by specifying a *null* query string.

Example 57–16 Retrieving a List of All Discoverable Spaces

```
List<String> allDiscoverableGroupSpaces = client.getDiscoverableGroupSpaces(null)
```

57.2.5.3.4 Retrieving Space Metadata Use the `getGroupSpaceMetadata` or `getGroupSpaceMetadataByGuid` APIs to obtain information (metadata) about a particular space. This includes information such as the description of the space, the name of the user who created it, the date on which it was last updated, and so on.

To use the `getGroupSpaceMetadata` API, specify the name of the space. To use the `getGroupSpaceMetadataByGuid` API, specify the GUID of the space. Note that while the space name may be changed during the existence of the space, the GUID always remains the same. You can obtain the GUID of a space as follows:

```
getGroupSpaceMetadata("spaceName").getGuid();
```

Both APIs return a bean object that contains more APIs that you can then use for retrieving the space metadata. These APIs are provided by the `GroupSpaceWSMetadata` class. Table 57–3 lists the APIs returned by the bean object:

Table 57–3 APIs for Retrieving Space Metadata

Spaces API	Description
`getCreatedBy`	Returns the name of the person who created the space.
`getSpaceAttributes`	Returns the custom attributes for a space.
`getDescription`	Returns the description of the space.
`getDisplayName`	Returns the display name of the space.
`getGroupSpaceState`	Returns the state of the space: active, offline, deleted.
`getGuid`	Returns the GUID of the space.
`getIconURL`	Returns the location of the space icon.
`getKeywords`	Returns a comma separated list of keywords used to describe the space.
`getLastUpdated`	Returns the date the space was last updated.
`getLogoURL`	Returns the location of the space logo.
`getMailingList`	Returns the mailing list for the space.
`getMembers`	Returns the current list of space members.
`getName`	Returns the internal name of the space.
`isDiscoverable`	Returns if the space is returned in searches made by users who are not members of the space.
`isPublic`	Returns if the space is available to users who are not logged in.

Example 57–17 retrieves the *Description*, *Keywords*, and *Last Updated Date* information for the *Databases* space given the space's name.

Example 57–17 Retrieving Space Metadata Using the Space Name

```
//get the exact name of the space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
```

```
//check that the API returns a single result
if(myGroupSpace.size() == 1)
{
  //retrieve the metadata
  GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
  //get space description
  System.out.println("Description: " + metadata.getDescription());
  //get space keywords
  System.out.println("Keywords: " + metadata.getKeywords());
  //get the date the space was last updated
  System.out.println("Last Updated Date: "+ metadata.getLastUpdated().toString());
}
```

Example 57–18 retrieves the *Display Name, Creator,* and *Last Updated Date* information for the *Databases* space given the space GUID.

Example 57–18 Retrieving Space Metadata Using Space GUID

```
GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadataByGuid("Guid");
//get space display name
System.out.println("Display Name: " + metadata.getDisplayName());
//get the name of the user who created the space
System.out.println("Created By: " + metadata.getCreatedBy());
//get the date the space was last updated
System.out.println("Last Updated Date: " + metadata.getLastUpdated().toString());
```

Example 57–19 retrieves the *Name, Description, Type,* and *Value* for every custom attribute associated with the *Databases* space.

Example 57–19 Retrieving Custom Attribute Metadata

```
//get the exact name of the space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
//check that the API returns a single result
if(myGroupSpace.size() == 1)
{
  //retrieve the metadata
  GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
  //get list of custom attributes
  List<GSCustomAttribute> attributes = metadata.getCustomAttributes();
  //get name, description, type, and value for each custom attribute
  for(GSCustomAttribute attribute: attributes)
  {
    System.out.println("Name :" + attribute.getName());
    System.out.println("Description :" + attribute.getDescription());
    System.out.println("Type :" + attribute.getType());
    System.out.println("value.toString() :" + attribute.getValue().toString());
  }
}
```

Example 57–20 retrieves a list of members of the *Databases* space.

Example 57–20 Retrieving Membership Information

```
//get the exact name of the space
List<String> myGroupSpace = client.getGroupSpaces("Databases");
//check that the API returns a single result
if(myGroupSpace.size() == 1)
{
  //retrieve the metadata
```

```
  GroupSpaceWSMetadata metadata = client.getGroupSpaceMetadata(myGroupSpace);
  //get the list of members
  for(String member: metadata.getMembers())
  {
    System.out.println("Member UID: " + member);
  }
}
```

57.2.5.3.5 Retrieving a List of Space Templates

Use the `getGroupSpaceTemplates` API to obtain a list of space templates that match a given query string. To use this API, specify a query string. A null value query string returns a list of *all* templates that are accessible to the current user.

The API returns templates that are accessible to the current user, up to a maximum of 500.

Example 57–21 returns a list of space templates containing the string *Interest*.

Example 57–21 Retrieving a List of Specific Space Templates

```
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates("Interest");
```

Example 57–22 returns a list of *all* space templates to which the current user has access. This is achieved by specifying a *null* query string.

Example 57–22 Retrieving a List of All Space Templates

```
List<String> allGroupSpaceTemplates = client.getGroupSpaceTemplates(null);
```

57.2.5.3.6 Retrieving Space Template Metadata

Use the `getGroupSpaceTemplateMetadata` and `getGroupSpaceTemplateMetadataByGuid` APIs to obtain information (metadata) about a particular space template. This includes information such as the description of the template, the name of the user who created it, and so on.

To use the `getGroupSpaceTemplateMetadata` API, specify the name of the template. To use the `getGroupSpaceTemplateMetadataByGuid` API, specify the GUID of the template. Note that while the space template name may be changed during the existence of the template, the GUID always remains the same.

Both APIs return a bean object that contains more APIs that you can then use for retrieving the space template metadata. These APIs are provided by the `GroupSpaceWSMetadata` class. Table 57–4 lists the APIs returned by the bean object:

Table 57–4 APIs for Retrieving Space Template Metadata

Spaces API	Description
`getCreatedBy`	Returns the name of the person who created the template.
`getDescription`	Returns the description of the template.
`getDisplayName`	Returns the display name of the template.
`getGuid`	Returns the GUID of the template.
`getIconURL`	Returns the location of the template icon.
`getKeywords`	Returns a comma separated list of keywords used to describe the template.
`getLogoURL`	Returns the location of the template logo.

Table 57–4 (Cont.) APIs for Retrieving Space Template Metadata

Spaces API	Description
`getName`	Returns the internal name of the template.

Example 57–23 retrieves the *GUID, Description,* and *Created By* information for the *CommunityofInterest* space template given the template name.

Example 57–23 Retrieving Template Metadata Using the Template Name

```
GroupSpaceWSMetadata metadata =
client.getGroupSpaceTemplateMetadata(myGroupSpaceTemplate);
//get the exact name of the space template
List<String> myGroupSpaceTemplate =
client.getGroupSpaceTemplates("CommunityofInterest");
//check that the API returns a single result
if(myGroupSpaceTemplate.size() == 1)
{
  //retrieve the metadata -- get GUID
  System.out.println("GUID: " + metadata.getGuid());
  //get template description
  System.out.println("Description: " + metadata.getDescription());
  //get name of user who created the template
  System.out.println("Keywords: " + metadata.getCreatedBy());
}
```

Example 57–24 retrieves the *name* of a space template given the template GUID.

Example 57–24 Retrieving Template Metadata Given the Template GUID

```
GroupSpaceWSMetadata templGuidMetadata =
client.getGroupSpaceTemplateMetadataByGuid("Guid");
System.out.println("Template Name: " + templGuidMetadata.getName());
```

57.2.5.3.7 Retrieving the Spaces URL Use the `getWebCenterSpacesURL` API to obtain the URL of the Spaces application.

Example 57–25 retrieves the URL of the currently running instance of Spaces.

Example 57–25 Retrieving the WebCenter Portal: Spaces URL

```
String myWebCenterSpacesURL = client.getWebCenterSpacesURL();
```

57.2.5.3.8 Retrieving a Space URL Use the `getGroupSpaceURL` API to obtain the URL of a specific space. This is useful for when you want to construct a hyperlink for a space or you have a relative URL that you need to make into an absolute URL. To use this API, specify the name of the space.

Example 57–26 retrieves the URL of the *Databases* space.

Example 57–26 Retrieving a Space URL

```
String myGroupSpaceURL = client.getGroupSpaceURL("Databases");
```

Example 57–27 prints a list of spaces as hyperlinks.

Example 57–27 Printing a List of Spaces as Hyperlinks

```
//get the list of spaces
List<String> spaces = client.getGroupSpaces("");
//print the list of spaces as hyperlinks
for (String spaceName : spaces)
{
  print("<a href =""" + client.getGroupSpaceURL(spaceName) + "">" + spaceName +
"</a><br>");
}
```

To construct the URL of a particular page in a space, retrieve the space URL and then add the page information. For information about how to create pretty URLs for space pages, see "Spaces Pretty URLs" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

57.2.5.3.9 Retrieving RSS Feed URLs for Space Services Space members can find out what is happening in a space through various RSS news feeds. The following space RSS feeds are available:

- Announcements RSS - View space announcements
- Discussions RSS - Track contributions to discussion forums
- Lists RSS - Watch for revisions to lists
- Recent Activity RSS - Monitor recent activities

You can retrieve the RSS feed URLs for these space services, from a WebCenter Portal application, using the following Spaces APIs:

- `getServiceRSSFeedURL`
- `getServiceRSSFeedURLbyGuid`

To obtain an RSS feed URL, you must identify the space (by name or GUID) and specify the service required (using a service ID). Service IDs are available as constants in `GroupSpaceWSClient` as follows:

- Announcements - `GroupSpaceWSClient.ANNOUNCEMENT_SERVICE_ID`
- Discussions - `GroupSpaceWSClient.DISCUSSION_FORUM_SERVICE_ID`
- Lists - `GroupSpaceWSClient.LIST_SERVICE_ID`
- Recent Activities - `GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID`

Retrieving Space RSS News Feed URLs Using `getServiceRSSFeedURL`

Use the `getServiceRSSFeedURL` API to obtain service-related RSS news feed URLs for a particular space by specifying the space name.

To retrieve the RSS news feed URL for a service using `getServiceRSSFeedURL`, use the following:

```
String service_URL = client.getServiceRSSFeedURL("groupspace_name",service_ID);
```

Where:

service_URL refers to the service parameter being retrieved

groupspace_name is the name of the space

service_ID is the ID of the service for which you want to retrieve the RSS news feed URL. Use one of the following: `ANNOUNCEMENT_SERVICE_ID`, `DISCUSSION_FORUM_SERVICE_ID`, `RECENT_ACTIVITY_SERVICE_ID`, or `LIST_SERVICE_ID`.

Therefore, depending on the service required, you can use the following:

- `String service_URL = client.getServiceRSSFeedURL("groupspace_name", GroupSpaceWSClient.ANNOUNCEMENT_SERVICE_ID)`
- `String service_URL = client.getServiceRSSFeedURL("groupspace_name", GroupSpaceWSClient.DISCUSSION_FORUM_SERVICE_ID)`
- `String service_URL = client.getServiceRSSFeedURL("groupspace_name", GroupSpaceWSClient.LIST_SERVICE_ID)`
- `String service_URL = client.getServiceRSSFeedURL("groupspace_name", GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID)`

Example 57–28 retrieves the recent activity RSS feed URL for a space named Finance_Project:

Example 57–28 Retrieving the RSS Feed URL for Recent Activity

```
String recentActivityURL = client.getServiceRSSFeedURL("Finance_Project",
GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID);
```

Retrieving RSS News Feed URLs Using `GetServiceRSSFeedURLbyGuid`

Use the `getServiceRSSFeedURLbyGuid` API to obtain service-related RSS news feed URLs for a particular space by specifying the space's GUID.

To retrieve the RSS news feed URL for a service using `getServiceRSSFeedURLbyGuid`, use the following:

```
String service_URL = client.getServiceRSSFeedURLbyGuid("groupspace_GUID",service_
ID);
```

Where:

service_URL refers to the service parameter being retrieved

groupspace_GUID is a space GUID. For information about obtaining a space's GUID, see Section 57.2.5.3.4, "Retrieving Space Metadata."

service_ID is the ID of the service for which you want to retrieve the RSS news feed URL. Use one of the following: `ANNOUNCEMENT_SERVICE_ID`, `DISCUSSION_FORUM_SERVICE_ID`, `RECENT_ACTIVITY_SERVICE_ID`, or `LIST_SERVICE_ID`.

Example 57–29 retrieves the recent activity RSS feed URL for a space with the GUID: s2201fa44_b441_4bdd_950e_47307f6f9800:

Example 57–29 Retrieving the RSS Feed URL for Recent Activity

```
String recentActivityURL = client.getServiceRSSFeedURLbyGuid("s2201fa44_b441_4bdd_
950e_47307f6f9800", GroupSpaceWSClient.RECENT_ACTIVITY_SERVICE_ID);
```

57.2.6 How to Handle Exceptions Raised by Spaces APIs

The `GroupSpaceWSException` class provides APIs for handling exceptions raised by Spaces APIs.

Table 57–5 Spaces APIs in the GroupSpaceWSException Class

API	Class	Description
`getLocalizedMessage`	GroupSpaceWSException	Composes the localized error message.
`printStackTrace`	GroupSpaceWSException	Prints the exception and its back trace to the standard error stream.

This section includes the following subsections:

- Providing Localized Error Messages
- Listing the Error Stack

57.2.6.1 Providing Localized Error Messages

Sometimes you may find that the default error messages provided by the APIs are not specific enough for your particular application. In these cases you can provide your own error messages.

Use the `getLocalizedMessage` API to compose application-specific error messages.

Example 57–30 shows a servlet that includes code to create a space. If any exceptions are raised during the creation process, a localized message is printed.

Example 57–30 Printing a Localized Error Message

```
servlet1.java

doGet()
{
...
  print("<b>Output</b>");
  try
    {
      GroupSpaceWSClient client = new GroupSpaceWSClient(context);
      client.createGroupSpace("Databases", "Databases, "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");
      print("Successfully created space");
    }
  catch(GroupSpaceWSException ex)
    {
      if(ex instanceof GroupSpaceNameNullException)
        print(ex.getLocalizedMessage());
      else if (ex instanceof GroupSpaceDescNullException)
        print(ex.getLocalizedMessage());
    }
...
}
```

57.2.6.2 Listing the Error Stack

For debugging purposes, it is often useful to see which errors ultimately led to the failure of a particular operation to discover the underlying cause of the problem. Use the `printStackTrace` API to list all the errors that caused a particular exception.

Example 57–31 prints the exception and all the errors leading up to it.

Example 57–31 Listing the Error Stack

servlet1.java

```
doGet()
{
...
  print("<b>Output</b>");
  try
    {
      GroupSpaceWSClient client = new GroupSpaceWSClient(context);
      client.createGroupSpace("Databases", "Databases", "A community for people
interested in databases", "databases, oracle", "CommunityofInterest");
      print("Successfully created space");
    }
  catch(GroupSpaceWSException ex)
    {
      ex.printStackTrace();
    }
...
}
```

57.2.7 Finding More Information on Spaces APIs

API Reference Documentation

For detailed syntax information for each API, see the *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

57.2.8 Troubleshooting Issues with Spaces APIs

If you experience issues with Spaces APIs, check the following:

1. Verify that the credential stores for both WebCenter Portal: Spaces and the client WebCenter Portal application are configured correctly.

   See "Updating the Credential Stores" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

2. Ensure the Spaces client context is set up correctly in the WebCenter Portal application by checking the alias passed in the `context.setRecipientKeyAlias`. The alias should be the public key alias of the producer (Spaces), for example:

   ```
   GroupspaceWSContext context = new GroupspaceWSContext();
   context.setEndPoint(endPointUrl);
   context.setRecipientKeyAlias("orakey");
   groupspaceInternalWSClient = new GroupspaceWSInternalClient(context);
   ```

   In this example, the public key alias of the producer is **orakey**. See also, Setting Up the Spaces Client Context.

3. Check that keystores exist at both ends of the connection. For example:

   - `webcenter.jks` (copied to Spaces end)

   - `clientapi.jks` (copied to WebCenter Portal application end)

   For example, the following commands generate `clientapi.jks` and `webcenter.jks` for a simple topology:

```
keytool -genkeypair -keyalg RSA -dname "cn=spaces,dc=example,dc=com" -alias
orakey -keypass mypassword -keystore webcenter.jks -storepass mypassword
-validity 360
keytool -exportcert -v -alias orakey -keystore webcenter.jks
-storepass mypassword -rfc -file webcenter.cer
keytool -importcert -alias webservice_client_api
-file webcenter.cer -keystore clientapi.jks -storepass mypassword
keytool -genkeypair -keyalg RSA -dname "cn=clientapi,dc=example,dc=com"
-alias clientapi -keypass mypassword -keystore clientapi.jks -storepass
mypassword -validity 360
keytool -exportcert -v -alias clientapi -keystore clientapi.jks -storepass
mypassword -rfc -file clientapi.cer
keytool -importcert -alias clientapi -file clientapi.cer -keystore
webcenter.jks -storepass mypassword
```

See also, Securing the Connection Between the Application and Spaces.

57.3 Using the WebCenter Portal: Spaces REST APIs

Oracle WebCenter Portal: Spaces provides REST APIs to support various space-related operations. You can use REST APIs to perform the following actions in the Spaces application:

- List all spaces in the Spaces application, list public spaces only, list joined spaces, list discoverable spaces.
- View details for a single space.
- List members of a space.
- List role assignments for space members.
- List custom attributes defined for a space.
- View the details of a particular custom attribute.
- View and add lists for a space.
- View and update list items (rows) for a list.
- View, update, and delete list item details (columns).

This section describes the WebCenter Portal: Spaces REST APIs. It contains the following subsections:

- Section 57.3.1, "WebCenter Portal: Spaces Entry Point"
- Section 57.3.2, "WebCenter Portal: Spaces Resource Type Taxonomy"
- Section 57.3.3, "WebCenter Portal: Spaces Security Considerations"
- Section 57.3.4, "WebCenter Portal: Spaces Resource Types"

For an introduction to the REST APIs, see Section 54, "Using Oracle WebCenter Portal REST APIs."

57.3.1 WebCenter Portal: Spaces Entry Point

Each REST service has a link element within the Resource Index that provides the entry point for that service. To find the entry point for WebCenter Portal: Spaces REST APIs, find the link element with a `resourceType` of:

```
urn:oracle:webcenter:spaces
```

The corresponding `href` or `template` element provides the URI entry point which returns a list of spaces accessible to the current user. The client sends HTTP requests to this entry point to work with WebCenter Portal: Spaces.

For more information about the Resource Index, see Section 54.5.1, "The Resource Index".

For more information about resource types, see Section 54.5.2.1, "Resource Type."

57.3.2 WebCenter Portal: Spaces Resource Type Taxonomy

When the client has identified the entry point, it can then navigate through the resource type taxonomy to perform the required operations. For more information about the individual resource types, see the appropriate section in Section 57.3.4, "WebCenter Portal: Spaces Resource Types".

The taxonomy for WebCenter Portal: Spaces is:

```
urn:oracle:webcenter:spaces
urn:oracle:webcenter:spaces:siteresources
urn:oracle:webcenter:spaces:resource:templates
urn:oracle:webcenter:spaces:resource:template
   urn:oracle:webcenter:space:attributes
   urn:oracle:webcenter:space:roles
   urn:oracle:webcenter:space
   urn:oracle:webcenter:space:icon
   urn:oracle:webcenter:space:members
      urn:oracle:webcenter:space:member
   urn:oracle:webcenter:space:attributes
      urn:oracle:webcenter:space:attribute
   urn:oracle:webcenter:space:roles
      urn:oracle:webcenter:space:role
   urn:oracle:webcenter:space:resourceindex
      urn:oracle:webcenter:space:lists
      urn:oracle:webcenter:space:list
         urn:oracle:webcenter:space:rows
            urn:oracle:webcenter:space:row
         urn:oracle:webcenter:space:columns
            urn:oracle:webcenter:space:column
```

57.3.3 WebCenter Portal: Spaces Security Considerations

There are no specific security considerations for this service. For general security considerations, see Section 54.8, "Security Considerations for WebCenter Portal REST APIs."

57.3.4 WebCenter Portal: Spaces Resource Types

The following sections provide all the information you need to know about each resource type:

- urn:oracle:webcenter:spaces
- urn:oracle:webcenter:siteresources
- urn:oracle:webcenter:spaces:resource:templates
- urn:oracle:webcenter:space
- urn:oracle:webcenter:space:resourceindex
- urn:oracle:webcenter:space:roles

- urn:oracle:webcenter:space:attributes
- urn:oracle:webcenter:space:attribute
- urn:oracle:webcenter:space:members
- urn:oracle:webcenter:space:member
- urn:oracle:webcenter:space:lists
- urn:oracle:webcenter:space:list
- urn:oracle:webcenter:space:list:rows
- urn:oracle:webcenter:space:list:row
- urn:oracle:webcenter:space:list:columns
- urn:oracle:webcenter:space:list:column

57.3.4.1 urn:oracle:webcenter:spaces

Use this resource type to identify the URI to use to view a list of spaces (`GET`) and to create a new space (`POST`). The response from a `GET` operation includes each space, and each space includes links to operate on that space.

Navigation Paths to spaces

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
```

Supported Methods for spaces

The following methods are supported by this resource:

- `GET` – Returns information on all spaces.
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `itemsPerPage`
    - `projection` - Allowed values are `summary` and `details`. The default is `summary`.
    - `visibility` - Determines which spaces are included in the list: all spaces, public spaces only, joined spaces, or discoverable spaces.

      Allowed values are `all`, `public`, `joined` and `discoverable`. The default is `joined`.
  - **response - body:** collection of spaces
- `POST` – Creates a new space.
  - **request - body:** space

```
<space>
   <name>aName</name>
   <description>aDescription</description>
   <templateName>aTemplate</templateName>
</space>
```

Note: The `templateName` parameter must be a valid template name returned from urn:oracle:webcenter:spaces:resource:templates. For example, the default template names are:

```
Basic
CommunityOfInterest
ProjectSpace
```

Resource Types Linked to from spaces

Table 57–6 lists the resource types that the client can link to from this resource.

Table 57–6 Related Resource Types for urn:oracle:webcenter:spaces

rel	resourceType
self	`urn:oracle:webcenter:spaces`
	`urn:oracle:webcenter:space`

57.3.4.2 urn:oracle:webcenter:siteresources

Use this resource type to identify the URI to use to retrieve a list of Site Resources (`GET`). The response from a `GET` operation includes a list of all site resource names. The main purpose of retrieving site resource names is to know what all resources is available in the space.

The returned list includes the resources that are found under the Resources tab of the Spaces application Administration console, and include resources like page templates, navigations, skins, page styles, and others.

Navigation Paths to siteresources

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      siteresources
```

Supported Methods for siteresources

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `itemsPerPage`
    - `projection` - Allowed values are `summary` and `details`. The default is `summary`.
    - `q` - You can search on `siteResourceType` and/or `seeded`.

      If `siteResourceType` is specified, the query returns all site resource names for that site resource type.

The format is: `q = siteResourceType:equals:<value>`

The possible values are `siteTemplate`, `contentPresenter`, `pageStyle`, `navigation`, `resourceCatalog`, or `skin`.

Example:

`q=siteResourceType:equals:siteTemplate`

If `seeded` is specified as `true`, the query returns all site resources names that are seeded.

The format is: `q=seeded:equals:<value>`.

The possible values are `true` or `false`.

Example:

`q=seeded:equals:false` (returns all non-seeded resources)

If a non-seeded site template is required, the query is:

```
siteResourceType:equals:siteTem-
plate;seeded:equals:false
```

– **response - body:** list of all site resource names for all the site resource types for the specified scope. If the `siteResourceType` is specified, returns all the site resource names for that site resource type.

Read-Only Elements for siteresources

Table 57–7 lists the read-only elements for siteresources.

Table 57–7 Read-Only Elements for urn:oracle:webcenter:siteresources

Element	Type	Description
guid	String	Global ID for the site resource.
displayName	String	Name of the site resource as displayed to end users.
name	String	The name of the site resource. A complete list of all of the site resources, by name, is provided in Table 57–8.
description	String	Description of the site resource.
siteResourceType	String	Type of the site resource.
scopeName	String	Either the name of the space or the default scope at the application level.
seeded	Boolean	Set to true if the site resource is pre-populated.
visible	Boolean	Set to true if this site resource is visible in the user interface.
createdBy	String	Username of the user who created the site resource.
createdDate	Date	Date the site resource was created.
modifiedBy	String	Username of the user who updated the site resource.
modifiedDate	Date	Date on which the site resource was modified.
version	String	Version of the site resource.

List of Site Resources by Name

Table 57–8 lists the site resources by name, and gives the type and description of each resource.

Table 57–8 List of Site Resources by Name Provided by WebCenter Portal

Name	Type	Description
Accordion View	contentPresenter	Displays multiple content items in an accordion (stacked) format. When an item is clicked, the other items collapse to reveal the selected items details.
Blank	pageStyle	Blank Page Style
Blank	taskFlowStyle	Blank
Blog	pageStyle	Blog Page Style
Bulleted View	contentPresenter	Displays multiple content items as a bulleted list. Only content items are displayed; sub-folders are omitted.
Bulleted with Folder Label View	contentPresenter	Displays multiple content items as a bulleted list. The name of the parent folder is displayed as a label above the list. Only content items are displayed; sub-folders are omitted.
Carousel View	contentPresenter	Displays multiple content items in a carousel format. Users browse through items using a slider bar from left to right.
Default Document Details View	contentPresenter	Displays detailed information about a single content item, including creation date, modification date, creator, user who last modified the item, and path, as well as any comments.
Default Home Space Catalog	resourceCatalog	Used when editing application-level pages and task flows in your Home space.
Default List Item View	contentPresenter	Used by a single content item view to display a single line with an icon and item name as a link that either displays or downloads the item when clicked.
Default List Item View for Folders	contentPresenter	Used by a single content item view to display a single line with an icon and item name as a link that either displays or downloads the item when clicked.
Default Page Template Catalog	resourceCatalog	Used when editing page templates
Default Space Catalog	resourceCatalog	Used when editing space-level pages and task flows
Default View	contentPresenter	Displays a single content item. Image and HTML content items are displayed directly in the browser. For other item types, details are displayed, along with a link to download the associated file.
Fusion FX	skin	
Fusion Side Navigation	siteTemplate	Stretching Page Layout with Side Navigation. Use Fusion FX Skin.
Fusion Top Navigation	siteTemplate	Stretching Page Layout with Top Navigation. Use Fusion FX Skin.

Table 57–8 (Cont.) List of Site Resources by Name Provided by WebCenter Portal

Name	Type	Description
Home Page	pageStyle	Home Page style
Icon View	contentPresenter	Displays multiple content items in a tiled format, using icons to represent sub-folders and content items.
Left Narrow	pageStyle	Left Narrow Page Style
List View	contentPresenter	Displays multiple content items in a simple list.
List with Details Panel View	contentPresenter	Displays multiple content items in a list on the left. A panel on the right displays the details of a selected item.
Parameter Form	taskFlowStyle	
Right Narrow	pageStyle	
Sortable Table View	contentPresenter	Displays multiple content items in a sortable table that includes the document name, date created, and date modified.
Html Page	pageStyle	HTML page style
Spaces Default Navigation Model	navigation	Navigation model that displays current pages in a space
Spaces Navigation with Blogs/Wikis/Lists Submenus	navigation	Navigation model that displays current pages and submenus for blogs, wikis and lists
Stretch	taskFlowStyle	
Tabbed View	contentPresenter	Displays multiple content items as tabs. Clicking a tab displays the associated item's details.
Three Column	pageStyle	Three Column Page Style
Web Page	pageStyle	Web Page Page Style
Public-Pages Template	siteTemplate	Default Page Template for Public pages.
Collaborative with Top Navigation	siteTemplate	Page Template for Collaborative Spaces, with Flowing Layout and Top Navigation. Use WebCenter Spaces FX Skin.
Spaces FX	skin	WebCenter Spaces FX Skin
Portal-centric with Top Navigation	siteTemplate	Page Template for Portal-centric Sites, with Flowing Layout and Top Navigation. Use WebCenter Spaces FX Skin.
Side Navigation	siteTemplate	Flowing Page Layout with Side Navigation. Use WebCenter Spaces FX Skin.
Side Navigation (Stretch)	siteTemplate	Stretching Page Layout with Side Navigation. Use WebCenter Spaces FX Skin.
Top Navigation	siteTemplate	Default Page Template with Flowing Layout and Top Navigation. Use WebCenter Spaces FX Skin.
Top Navigation (Stretch)	siteTemplate	Stretching Page Layout with Top Navigation. Use WebCenter Spaces FX Skin.
Wiki	pageStyle	Wiki Page Style

Writeable Elements for siteresources

There are no writeable elements for this resource.

57.3.4.3 urn:oracle:webcenter:spaces:resource:templates

Use this resource type to identify the URI to use to retrieve a list of space templates (`GET`). The response from a `GET` operation includes a list of template names. The main purpose of retrieving template names is to provide a value for the `templateName` parameter for the create space command.

Navigation Paths to templates

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resource
         templates
```

Supported Methods for templates

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `projection` - Allowed values are `summary` and `details`. The default is `summary`.
    - `q` - Performs searches.
    - `itemsPerPage`
  - **response - body:** list of template names

For more information about query parameters, see Section 54.5.2.5, "Templates."

Resource Types Linked to from templates

Table 57–9 lists the resource types that the client can link to from this resource.

Table 57–9 Related Resource Types for urn:oracle:webcenter:spaces

rel	resourceType
self	`urn:oracle:webcenter:spaces:resource:templates`

57.3.4.4 urn:oracle:webcenter:space

Use this resource type to identify the URI to use to view details for a single space (`GET`), create a nested space (`POST`), or delete a space (`DELETE`). The response from a `GET` operation includes the specific space identified by the URI, including its creator, description, members, and custom attributes.

> **Note:** The response from this resource also includes a link to the space's message board.

Navigation Paths to space

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
```

Supported Methods for space

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:**
    - `projection` - Allowed values are `summary`, and `details`. The default is `details`. Choose `summary` to exclude member and custom attribute details.
  - **response - body:** Space
- `POST` – Creates a new space that is a child of the named space (a subspace). This feature supports creating hierarchical or nested spaces.
  - **request - body:** Space

```
<space>
   <name>FooSpace</name>
   <description>foobar</description>
   <templateName>CommunityOfInterest</templateName>
</space>
```

> **Note:** The `templateName` parameter must be a valid template name returned from urn:oracle:webcenter:spaces:resource:templates. For example, the default template names are:
>
> ```
> Basic
> CommunityOfInterest
> ProjectSpace
> ```

- `DELETE` – Deletes the named space.
  - **request - body:** None

    On delete, a 204 status code is returned whether or not the space existed.

For more information about query parameters, see Section 54.5.2.5, "Templates."

Writable Elements for space

Table 57–10 lists the writable elements for this resource.

Table 57–10 Writable Elements for urn:oracle:webcenter:space

Element	Type	Description
`description`	String	Description of the space.
`name`	String	Internal name for the space.

Table 57–10 (Cont.) Writable Elements for urn:oracle:webcenter:space

Element	Type	Description

`templateName`	String	Must be a valid template name returned from urn:oracle:webcenter:spaces:resource:templates. For example, the default template names are:
		`Basic`
		`CommunityOfInterest`
		`ProjectSpace`

Read-only Elements for space

Table 57–11 lists the read-only elements for this resource.

Table 57–11 Read-only Elements for urn:oracle:webcenter:space

Element	Type	Description
`guid`	String	Global ID for the space.
`creator`	String	ID of the user who created the space.
`author`	personReference	User information about the user who created the space, including GUID, ID, display name, and a link to the profile icon (same user as `creator`).
`displayName`	String	Name of the space as displayed to members.
`iconUrl`	String	Fully qualified URL for the space's icon image.
`logoUrl`	String	Relative URL for the space's logo image.
`isOffline`	Boolean	Indicates whether the space is offline or online.
`isDiscoverable`	Boolean	Specifies whether the space is discoverable.
`isPublic`	Boolean	Specifies whether the space is public.
`keywords`	String	Lists keywords for the space.
`mailingList`	String	Lists the mailing list email for the space if it has one.
`parentDisplayName`	String	Shows the parent display name if the space is a child space and has a parent. In this case, there will also be an `isInheritFromParent` boolean element that signifies whether the permissions of the parent space are inherited by the child space or not. Specifically, members of the parent space will be inherited.
`serviceDataCopied`	String	List of service ids whose data is to be copied to new spaces/templates when created based off this space.
`defaultLanguage`	String	The default language.
`landingPage`	String	URI to the space landing page if specified.

Table 57–11 (Cont.) Read-only Elements for urn:oracle:webcenter:space

Element	Type	Description
`isTemplate`	Boolean	Specifies whether or not this is a Template.
`isUnsubscriptionApprovalRequired`	Boolean	Specifies whether or not you need approval to unsubscribe.
`isPublishRss`	Boolean	Specifies whether or not the space has an associated RSS feed.
`isClosed`	Boolean	Specifies whether or not the space is closed.
`isBlockAllAccess`	Boolean	Specifies whether or not all access to space is blocked (not sure how this happens).
`members`	String	Returns the members element for the space.
`attributes`	String	Returns the attributes element for the space.

Resource Types Linked to from space

Table 57–12 lists the resource types that the client can link to from this resource.

Table 57–12 Related Resource Types for urn:oracle:webcenter:space

rel	resourceType
self	`urn:oracle:webcenter:space`
	`urn:oracle:webcenter:space:attributes`
	`urn:oracle:webcenter:space:members`
	`urn:oracle:webcenter:space:roles`
	`urn:oracle:webcenter:space:icon`
	`urn:oracle:webcenter:activities:stream`
	`urn:oracle:webcenter:messageBoard`

57.3.4.5 urn:oracle:webcenter:space:resourceindex

Use this resource type to identify the URI to use to view the Spaces resource index.

Navigation Paths to resourceindex

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
```

Supported Methods for resourceindex

The following methods are supported by this resource:

- `GET`
  - **request - body:** None

- **query parameters:** None
- **response - body:** resource index

Resource Types Linked to From resourceindex

Table 57–13 lists the resource types that the client can link to from this resource.

Table 57–13 Related Resource Types for urn:oracle:webcenter:space:resourceindex

rel	resourceType
	urn:oracle:webcenter:space:lists
	urn:oracle:webcenter:discussions:forum

57.3.4.6 urn:oracle:webcenter:space:roles

Use this resource type to get a list of roles for the named space (GET). The results returned from this call can be used to add a new member to a space with urn:oracle:webcenter:space:members.

Navigation Paths to space:roles

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         roles
```

Supported Methods for roles

The following methods are supported by this resource:

- GET – Returns the list of roles assigned to the named space.

Resource Types Linked to from roles

Table 57–14 lists the resource types that the client can link to from this resource.

Table 57–14 Related Resource Types for urn:oracle:webcenter:spaces:roles

rel	resourceType
self	urn:oracle:webcenter:space:roles
	urn:oracle:webcenter:space:role
urn:oracle:webcenter:parent	urn:oracle:webcenter:space

57.3.4.7 urn:oracle:webcenter:space:role

Use this resource type to get a named role (GET).

Use this resource type to get a list of roles for the named space (GET). The results returned from this call can be used to add a new member to a space with urn:oracle:webcenter:space:members.

Navigation Paths to space:role

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         role
```

Supported Methods for role

The following methods are supported by this resource:

- `GET` – Returns the named role.

Resource Types Linked to from role

Table 57–14 lists the resource types that the client can link to from this resource.

Table 57–15 Related Resource Types for urn:oracle:webcenter:spaces:roles

rel	resourceType
self	`urn:oracle:webcenter:space:role`
	urn:oracle:webcenter:space:role
urn:oracle:webcenter:parent	`urn:oracle:webcenter:space`

57.3.4.8 urn:oracle:webcenter:space:icon

Use this resource type to get the icon used for the named space (`GET`).

> **Note:** This resource type is used for space references, not for space responses. This resource can be used anywhere a space is referenced, like in an activity stream or a space event.

Navigation Paths to space:icon

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         icon
```

Supported Methods for icon

The following methods are supported by this resource:

- `GET` – Returns the icon used for the named space.

Resource Types Linked to from icon

Table 57–14 lists the resource types that the client can link to from this resource.

Table 57–16 Related Resource Types for urn:oracle:webcenter:spaces:roles

rel	resourceType
self	`urn:oracle:webcenter:space:icon`
urn:oracle:webcenter:parent	`urn:oracle:webcenter:space`

57.3.4.9 urn:oracle:webcenter:space:attributes

Use this resource type to identify the URI to use to view a list of custom attributes defined for a space (`GET`). The response from a `GET` operation includes each custom attribute for the space, and each attribute includes links to operate on that attribute. Also use this resource type to add custom attributes to a space (`POST`).

Navigation Paths to attributes

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         attributes
```

Supported Methods for attributes

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** collection of attributes
- `POST` – Adds a custom attribute to the named space.
  - **request - body:**

```
<attribute>
   <name>anAttribute</name>
   <value>aValue</value>
</attribute>
```

Writeable Elements for attributes

Table 57–17 lists the writeable elements for this resource.

Table 57–17 Writeable Resources for space:attributes

Element	Type	Description
name	String	Attribute name.
value	String	Attribute value.

Resource Types Linked to From attributes

Table 57–18 lists the resource types that the client can link to from this resource.

Table 57–18 Related Resource Types for urn:oracle:webcenter:space:attributes

rel	resourceType
self	urn:oracle:webcenter:space:attributes
	urn:oracle:webcenter:space:attribute
urn:oracle:webcenter:parent	urn:oracle:webcenter:space

57.3.4.10 urn:oracle:webcenter:space:attribute

Use this resource type to identify the URI to use to view the name and value of single space attribute (`GET`). The response from a `GET` operation includes the specific attribute identified by the URI.

Navigation Paths to attribute

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         attributes
```

Supported Methods for attribute

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** attribute

Read-only Elements for attribute

Table 57–19 lists the read-only elements for this resource.

Table 57–19 Read-only Elements for urn:oracle:webcenter:space:attribute

Element	Type	Description
name	String	Name of the custom attribute.
description	String	Description of the custom attribute.
type	String	Data type of the custom attribute.
value	String	Value of the custom attribute.

Resource Types Linked to From attribute

Table 57–20 lists the resource types that the client can link to from this resource.

Table 57–20 Related Resource Types for urn:oracle:webcenter:space:attribute

rel	resourceType
self	`urn:oracle:webcenter:space:attribute`
urn:oracle:webcenter:parent	`urn:oracle:webcenter:space`

57.3.4.11 urn:oracle:webcenter:space:members

Use this resource type to identify the URI to use to view a list of members for the space (`GET`). The response from a `GET` operation includes each member of the space, each member includes links to operate on that member.

Navigation Paths to members

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
```

Supported Methods for members

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** collection of members
- `POST` – Adds a member to the space.
  - **request - body:** member

```
<member>
    <name>vicki</name>
    <role>Participant</role>
</member>
```

- `DELETE` – Removes the named member from the space. You can obtain the member guid with urn:oracle:webcenter:space:member.
  - **request - body:** None

```
DELETE
http://hostname:portnum/rest/api/spaces/spacename/members/member_guid
```

Writeable Elements of spaces:space:members

Table 57–21 Writeable Elements of spaces:space:members

Element	Type	Description
name	String	The member's name.
role	String	The member's role. Obtain this value from urn:oracle:webcenter:space:roles.

Resource Types Linked to From members

Table 57–22 lists the resource types that the client can link to from this resource.

Table 57–22 Related Resource Types for urn:oracle:webcenter:space:members

rel	resourceType
self	`urn:oracle:webcenter:space:members`
	`urn:oracle:webcenter:space:member`
urn:oracle:webcenter:parent	`urn:oracle:webcenter:space`

57.3.4.12 urn:oracle:webcenter:space:member

Use this resource type to identify the URI to use to view details for a space member and their current role assignments (`GET`). The response from a `GET` operation includes the specific member identified by the URI, including a hyperlink to the member's profile.

Navigation Paths to member

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      space
         members
```

Supported Methods for member

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** member

Read-only Elements for member

Table 57–23 lists the read-only elements for this resource.

Table 57–23 Read-only Elements for urn:oracle:webcenter:space:member

Element	Type	Description
guid	String	Global ID for the space member.
name	String	User ID of the space member.
role	String	Role assigned to the member.
displayName	String	Display name of the member.

Resource Types Linked to From member

Table 57–24 lists the resource types that the client can link to from this resource.

Table 57–24 Related Resource Types for urn:oracle:webcenter:space:member

rel	resourceType
self	`urn:oracle:webcenter:space:member`
	`urn:oracle:webcenter:people:person`
urn:oracle:webcenter:people:icon	`urn:oracle:webcenter:people:person`

57.3.4.13 urn:oracle:webcenter:space:lists

Use this resource type to identify the URI to use to view (`GET`) and add (`POST`) lists for the space. The response from a `GET` operation includes each list in the space, and each list includes links used to operate on that list. The response from a `POST` operation includes the list that was created in this collection of lists and a link to operate on that list.

Navigation Paths to lists

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
      space
         lists
```

Supported Methods for lists

The following methods are supported by this resource:

- GET
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `itemsPerPage`
    - `q`

      You can search on `name`, `title`, `description`, `creator`, `created`, `modifier`, and `modified`.

      For string type elements (that is, `name`, `title`, `description`, `creator`, and `modifier`), you can use the following supported operands: `equals`, `not.equals`, and `contains`. For example:

      ```
      q=name:contains:issues
      q=creator:equals:monty
      ```

      For date type elements (that is, `created` and `modified`), you can use the following supported operands: `equals`, `not.equals`, `greater.than`, `greater.than.or.equals`, `less.than`, `less.than.or.equals`. For example:

      ```
      q=created:greater.than:10-SEP-2009
      ```

    - `projection` - Allowed values are `summary` and `details`. The default is `summary`.
  - **response - body:** collection of lists
- POST
  - **request - body:** list
  - **response - body:** list

For more information about query parameters, see Section 54.5.2.5, "Templates."

Resource Types Linked to From lists

Table 57–25 lists the resource types that the client can link to from this resource.

Table 57–25 Related Resource Types for urn:oracle:webcenter:space:lists

rel	resourceType
self	`urn:oracle:webcenter:space:lists`

Table 57–25 (Cont.) Related Resource Types for urn:oracle:webcenter:space:lists

rel	resourceType
	urn:oracle:webcenter:space:list

57.3.4.14 urn:oracle:webcenter:space:list

Use this resource type to identify the URI to use to view (GET), update (PUT), and delete (DELETE) a list belonging to a space. The response from a GET operation includes the specific list identified by the URI. The response from a PUT operation includes the modified version of the list identified by the URI. The response to a DELETE operation is a 204 status code.

Navigation Paths to list

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
         lists
             list
```

Supported Methods for list

The following methods are supported by this resource:

- GET
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** list
- PUT
  - **request - body:** list
  - **response - body:** list
- DELETE
  - **request - body:** None
  - **response - body:** None

Writable Elements for list

Table 57–26 lists the writable elements for this resource.

Table 57–26 Writable Elements for urn:oracle:webcenter:space:list

Element	Type	Required	Constraints	Description
name	String	Yes	1 or more characters	Name of the space list.
description	String	No	0 or more characters	Description of the list.
columns	urn:oracle:webcenter:space:list:columns	Yes	1 or more characters	Columns that make up the list.

Read-only Elements for list

Table 57–27 lists the read-only elements for this resource.

Table 57–27 Read-only Elements for urn:oracle:webcenter:list

Element	Type	Description
id	String	ID of the list.
name	String	Name of the list.
description	String	Description of the list.
scopeguid	String	Global ID of the parent space.
scopename	String	Name of the parent space.
creator	String	ID of the user that created the list.
author	personReference	User information about the user that created the list, including GUID, ID, display name, and a link to the profile icon (same user as `creator`).
created	Date	Date the list was created.
modifier	String	ID of the user that last modified the list.
modifiedByUser	personReference	User information about the user that last modified the list, including GUID, ID, display name, and a link to the profile icon (same user as `modifier`).
modified	Date	Date the list was last modified.
columns	String	Columns that make up the list.

Resource Types Linked to From list

Table 57–28 lists the resource types that the client can link to from this resource.

Table 57–28 Related Resource Types for urn:oracle:webcenter:space:list

rel	resourceType
self	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:rows`
	`urn:oracle:webcenter:space:list:columns`

57.3.4.15 urn:oracle:webcenter:space:list:rows

Use this resource type to identify the URI to use to view (`GET`) and create (`POST`) list rows (list items). The response from a `GET` operation includes each row in this list, each row includes links to operate on that row. The response from a `POST` operation includes the row that was created in this list and a link to operate on that row.

Navigation Paths to rows

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
        lists
           list
```

```
rows
```

Supported Methods for rows

The following methods are supported by this resource:

- GET
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `itemsPerPage`
    - `q`

      You can search on `name`, `title`, `description`, `creator`, `created`, `modifier`, and `modified`.

      For string type elements (that is, `name`, `title`, `description`, `creator`, and `modifier`), you can use the following supported operands: `equals`, `not.equals`, `contains`, and `starts.with`. For example:

      ```
      q=name:contains:issues
      q=creator:equals:monty
      ```

      For date type elements (that is, `created` and `modified`), you can use the following supported operands: `equals`, `not.equals`, `greater.than`, `greater.than.or.equals`, `less.than`, `less.than.or.equals`. For example:

      ```
      q=created:greater.than:10-SEP-2009
      ```
  - **response - body:** collection of rows
- POST
  - **request - body:** row
  - **response - body:** row

For more information about query parameters, see Section 54.5.2.5, "Templates."

Resource Types Linked to From rows

Table 57–29 lists the resource types that the client can link to from this resource.

Table 57–29 Related Resource Types for urn:oracle:webcenter:space:list:rows

rel	resourceType
self	`urn:oracle:webcenter:space:list:rows`
parent	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:row`

57.3.4.16 urn:oracle:webcenter:space:list:row

Use this resource type to identify the URI to use to view (`GET`), update (`PUT`), and delete (`DELETE`) a list row (list item). The response to a `GET` operation includes the specific row identified by the URI. The response to a `PUT` operation includes the modified version of the row identified by the URI. The response to a `DELETE` operation is a 204 status code.

Navigation Paths to row

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
         lists
            list
              rows
                 row
```

Supported Methods for row

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:** None
  - **response - body:** row
- `PUT`
  - **request - body:** row
  - **response - body:** row
- `DELETE`
  - **request - body:** None
  - **response - body:** None

For more information about query parameters, see Section 54.5.2.5, "Templates."

Writable Elements for row

Table 57–30 lists the writable elements for this resource.

Table 57–30 Writable Elements for urn:oracle:webcenter:space:list:row

Element	Type	Required	Constraints	Description
columns.column.id	String	Yes	1 or more characters	ID of the column (containing list item detail).
columns.column.name	String	No	1 or more characters	Name of the column.
columns.column.value	String	Yes	1 or more characters	Value of the column.

Read-only Elements for row

Table 57–31 lists the read-only elements for this resource.

Table 57–31 Read-only Elements for urn:oracle:webcenter:space:list:row

Element	Type	Description
id	String	ID of the row (list item).
listId	String	ID of the list.

Table 57–31 (Cont.) Read-only Elements for urn:oracle:webcenter:space:list:row

Element	Type	Description
scope	String	Global ID of the parent space.
creator	String	ID of the user that created the list item.
author	personReference	User information about the user that created the list item, including GUID, ID, display name, and a link to the profile icon (same user as `creator`).
created	Date	Date the list item was created.
modifier	String	ID of the user that last modified the list item.
modifiedByUser	personReference	User information about the user that last modified the list item, including GUID, ID, display name, and a link to the profile icon (same user as `modifier`).
modified	Date	Date the list item was last modified.

Resource Types Linked to From row

Table 57–32 lists the resource types that the client can link to from this resource.

Table 57–32 Related Resource Types for urn:oracle:webcenter:space:list:row

rel	resourceType
self	`urn:oracle:webcenter:space:list:row`

57.3.4.17 urn:oracle:webcenter:space:list:columns

Use this resource type to identify the URI to use to view (`GET`) and create (`POST`) list columns (list item detail). The response from a `GET` operation includes each column in this list, each column includes link to operate on that column. The response from a `POST` operation includes the column created in this list and a link to operate on that column.

Navigation Paths to columns

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
        lists
           list
              columns
```

Supported Methods for columns

The following methods are supported by this resource:

- `GET`
  - **request - body:** None
  - **query parameters:**
    - `startIndex`
    - `itemsPerPage`

        * `q` - Search parameters `STARTS.WITH` and `END.WITH` are not supported.
    - **response - body:** collection of columns
- `POST`
    - **request - body:** column
    - **response - body:** column

For more information about query parameters, see Section 54.5.2.5, "Templates."

Resource Types Linked to From columns

Table 57–33 lists the resource types that the client can link to from this resource.

Table 57–33 Related Resource Types for urn:oracle:webcenter:space:list:columns

rel	resourceType
self	`urn:oracle:webcenter:space:list:columns`
parent	`urn:oracle:webcenter:space:list`
	`urn:oracle:webcenter:space:list:column`

57.3.4.18 urn:oracle:webcenter:space:list:column

Use this resource type to identify the URI to use to view (`GET`), update (`PUT`), and delete (`DELETE`) a list column (list item detail). The response from a `GET` operation includes the specific column identified by the URI. The response from a `PUT` operation includes the modified version of the column identified by the URI. The response from a `DELETE` operation is a 204 status code.

Navigation Paths to column

This section shows how the client can navigate through the hypermedia to access this resource:

```
resourceindex
   spaces
      resourceindex
         lists
            list
               columns
                 column
```

Supported Methods for column

The following methods are supported by this resource:

- `GET`
    - **request - body:** None
    - **query parameters:** None
    - **response - body:** column
- `PUT`
    - **request - body:** column
    - **response - body:** column
- `DELETE`

- **request - body:** None
- **response - body:** None

Writable Elements for column

Table 57–34 lists the writable elements for this resource.

Table 57–34 Writable Elements for urn:oracle:webcenter:space:list:column

Element	Type	Required	Constraints	Description
columns.column.id	String	Yes	1 or more characters	ID of the column (containing list item detail).
columns.column.name	String	No	1 or more characters	Name of the column.
columns.column.value	String	Yes	1 or more characters	Value of the column.

Read-only Elements for column

Table 57–35 lists the read-only elements for this resource.

Table 57–35 Read-only Elements for urn:oracle:webcenter:space:list:column

Element	Type	Description
id	String	ID of the column (list item detail).

Resource Types Linked to From column

Table 57–28 lists the resource types that the client can link to from this resource.

Table 57–36 Related Resource Types for urn:oracle:webcenter:space:list:column

rel	resourceType
self	`urn:oracle:webcenter:space:list:column`

57.4 Exposing Enterprise Applications in Spaces

You can expose various enterprise applications inside the Spaces application, including:

- WebCenter Portal applications—applications developed using the WebCenter Portal: Framework.
- Other portals in the WebCenter Suite—for example, Oracle Portal.
- Oracle Applications Unlimited products—for example, E-Business Suite, Siebel, PeopleSoft, and JDEdwards.
- Other custom applications—applications developed using non-Oracle platforms.

Technologies such as WSRP, Oracle JSF Portlet Bridge, ADF task flows, and WebCenter Portal support for external applications, enable the Spaces application to consume and present application data in a unified way. The following sections tell you how.

This section includes the following subsections:

- Exposing WebCenter Portal Applications in Spaces
- Exposing Oracle Applications in Spaces

- Exposing Non-Oracle Applications in Spaces

57.4.1 Exposing WebCenter Portal Applications in Spaces

You can expose WebCenter Portal: Framework applications, built using the WebCenter Portal: Framework, as portlets and task flows in the Spaces application:

- **Portlets:** Web components that are deployed inside a container and generate dynamic content. For more information, see Chapter 58, "Overview of Portlets."
- **Oracle JSF Portlet Bridge:** A technology that enables you to easily expose existing ADF applications and task flows as JSR 168 portlets. For more information, see Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge."
- **Task flows:** ADF navigational components that define the transition between states and activities using call methods on managed beans, evaluating an EL expression, and so on. For more information see, "Getting Started with ADF Task Flows" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

57.4.2 Exposing Oracle Applications in Spaces

You can expose Oracle Applications Unlimited products, for example, E-Business Suite, Siebel, PeopleSoft, and JDEdwards, in the Spaces application, using:

- **Web services and ADF:** Enterprise applications like Siebel expose various web service interfaces. You can leverage these web service interfaces to build data controls, ADF task flows, and portlets that can be consumed in Spaces.
- **Prebuilt portlets:** Enterprise applications such as Oracle E-Business Suite ship portlets out of the box that you can register with Spaces and consume in the same way as any other portlet.

57.4.3 Exposing Non-Oracle Applications in Spaces

You can expose Web applications developed using non-Oracle platforms in the Spaces application, as follows:

- **Support for external applications:** External applications enable credential stores to pass mapped user identities to web applications that require their own authentication. For more information, see "Registering External Applications" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* and "Working with External Applications" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- **Composer iframe-level integration:** The Composer *Web Page* page style and component displays any Web page content, including pages from applications built using a different Web technology. For more information, see "Adding a View into Another Web Page to a Page" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- **Raw HTML markup injection in Composer:** The Composer *HTML Markup* layout component injects HTML or JavaScript snippets into pages in Spaces. You can use this component to display content from an external source. For more information see "Adding Markup, Such as HTML, JavaScript, and EL, to a Page" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- **OmniPortlet:** A portlet that publishes data from various data sources using a variety of layouts. For more information, see Chapter 65, "Creating Portlets with OmniPortlet."

Part X

Working with Portlets and Portals

Part X contains the following chapters:

- Section 58, "Overview of Portlets"
- Section 59, "Creating Portlets with the Oracle JSF Portlet Bridge"
- Section 60, "Creating Portlets with the Portlet Wizard"
- Section 61, "Coding Portlets"
- Section 62, "Testing and Deploying Your Portlets"
- Chapter 63, "Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer"
- Section 64, "Consuming Portlets"
- Section 65, "Creating Portlets with OmniPortlet"
- Section 66, "Creating Content-Based Portlets with Web Clipping"

58

Overview of Portlets

This chapter provides an overview of portlets, and describes, with the help of examples, the use of portlets. It explains portlet anatomy and the resources to create portlets. This chapter also explains the features, technologies, and tools to help you decide which portlet building technology best suits your needs.

This chapter includes the following sections:

- Section 58.1, "Introduction to Portlets"
- Section 58.2, "Portlet Technologies Matrix"

58.1 Introduction to Portlets

A portlet is a reusable web component that can draw content from many different sources. Figure 58–1 illustrates the Products portlet, a portlet that shows the products available on a shopping web site.

Figure 58–1 The Products Portlet



Portlets provide a means of presenting data from multiple sources in a meaningful and related way. Portlets can display excerpts of other web sites, generate summaries of key information, perform searches, and access assembled collections of information from a variety of data sources. Because different portlets can be placed on a common page, the user receives a single-source experience. In reality, the content may be derived from multiple sources.

A portlet may or may not be rendered in an inline frame (IFRAME). Inline frames enable the placement of a document within a rectangular region that includes scroll bars and borders.

Note: For more information about portlets and inline frames, see Section 64.4.12, "What You May Need to Know About IFRAMEs."

Within this IFRAME, portlets can display many types of content, including HTML, formatted text, images, or elements of an HTML form.

This section includes the following subsections:

- Section 58.1.1, "Portlet Anatomy"
- Section 58.1.2, "Portlet Resources"
- Section 58.1.3, "Portlet Modes"

58.1.1 Portlet Anatomy

Portlet anatomy is the visual representation of the portlet on a page. Figure 58–2 illustrates a typical portlet anatomy on a page in a Framework application. Note that the same portlet displayed in a different application could look different.

Figure 58–2 The Sample Lottery Portlet



What is rendered on the page is controlled not only by the portlet's own logic, but by the attributes of the portlet tag that binds the portlet to the page. Values for these attributes are specified at the design time of the application that consumes the portlet, rather than through the portlet's own logic.

Note: For information about attributes of the `adfp:portlet` tag, see Section 64.4, "Setting Attribute Values for the Portlet Tag."

For example, at application design time you can specify through portlet tag attributes that the runtime portlet should display a header and that its border should be of a specified thickness and color. In the header, you can include a portlet title and an Actions menu icon. The Actions menu icon is displayed on a portlet only when the

portlet header is displayed. If you choose not to display a header, then the Actions menu is displayed on a FadeIn-FadeOut toolbar that renders on a mouse rollover.

These elements are sometimes referred to as *portlet chrome*. The appearance of portlet chrome can be controlled through a style sheet and through style-related attributes of the `adfp:portlet` tag. The values of style-related attributes take precedence over styles specified through a style sheet, or *skin*.

> **Note:** For information about skins and style-related attributes, see Section 22.10, "Applying Styles to Components."

Through portlet tag attributes, you can include or omit display commands on the Actions menu. Actions menu items controlled through portlet tag attributes include *Maximize* and *Restore*. Maximize causes the maximized portlet to displace all other displayed portlets. These are displayed again when a user selects Restore.

> **Note:** The Maximize attribute is meaningful for a portlet only when the portlet is placed inside a `PanelCustomizable` core customizable component.

Other Actions menu items controlled through portlet tag attributes include the display or omission of the mode settings that were specified when the portlet was developed. If the portlet was built without including additional modes, then these commands do not appear on the Actions menu even when you indicate that they should through portlet tag attributes. In other words, portlet tag attributes are sometimes on/off switches that enable or disable the portlet's own built-in functionality.

Mode settings that appear on the Actions menu include such modes as *About*, and *Help*.

Users click the Personalize icon to alter their personal view of the portlet. The Personalize icon is displayed in the portlet header only to authenticated users (that is, users who are logged in). It does not display to Public or unauthenticated users. You must implement some form of application security for users to be able to personalize their portlet views.

> **Note:** If you are a developer creating portlets, and you want to test the Personalize mode without creating a complete security model for your application, see Section 69.12, "Configuring Basic Authentication for Testing Portlet Personalization."

Customization enables application administrators to edit a portlet's default settings at runtime. All users see the results of a customization.

Note: A typical customization setting is Portlet Title. At runtime, the portlet administrator can determine what title should appear in the portlet header. The Portlet Title can also be set at design time through portlet properties, using the `text` attribute of the `adfp:portlet` tag. Consider however that supplying a value to the `text` attribute at design time prevents customization of the Portlet Title at runtime.

For additional information about portlet tag attributes, see Section 64.4, "Setting Attribute Values for the Portlet Tag."

58.1.2 Portlet Resources

Portlet resources include the many prebuilt portlets available out of the box from many sources, including Oracle Portal, Oracle E-Business Suite, and third-party sources. Portlet resources also include programmatic portlets built through WebCenter Portal: Framework's JSR 286 (standards-based) and Oracle PDK-Java wizards, and through other portlet-building tools. Each of these tools offers different product features that are targeted toward different developer roles.

For specific information about each tool and its benefits, see Section 58.2, "Portlet Technologies Matrix."

This section includes the following subsections:

- Section 58.1.2.1, "JSF Portlets"
- Section 58.1.2.2, "Prebuilt Portlets"
- Section 58.1.2.3, "Parameter Form and Parameter Display Portlets"
- Section 58.1.2.4, "Web Clipping"
- Section 58.1.2.5, "OmniPortlet"
- Section 58.1.2.6, "Programmatic Portlets"
- Section 58.1.2.7, "Deciding Which Tool to Use"

58.1.2.1 JSF Portlets

What Are They?

JSF portlets are created using the Oracle JSF Portlet Bridge. The Oracle JSF Portlet Bridge enables application developers to expose their existing JSF applications and task flows as JSR 286 portlets. The Oracle JSF Portlet Bridge simplifies the integration of JSF applications with WSRP portlet consumers, such as Oracle Portal. You can create JSF portlets using the JSR 286 Java Portlet Wizard by invoking it from the New Gallery.

JSF portlets do not require separate source code from that of the JSF application. Since these portlets are created using the Oracle JSF Portlet Bridge, you need only maintain one source for both your application and your portlets. Similarly, when you deploy your JSF application, JSF portlets are also deployed with it. Therefore, using the bridge eliminates the requirement to store, maintain, and deploy your portlets separately from your application.

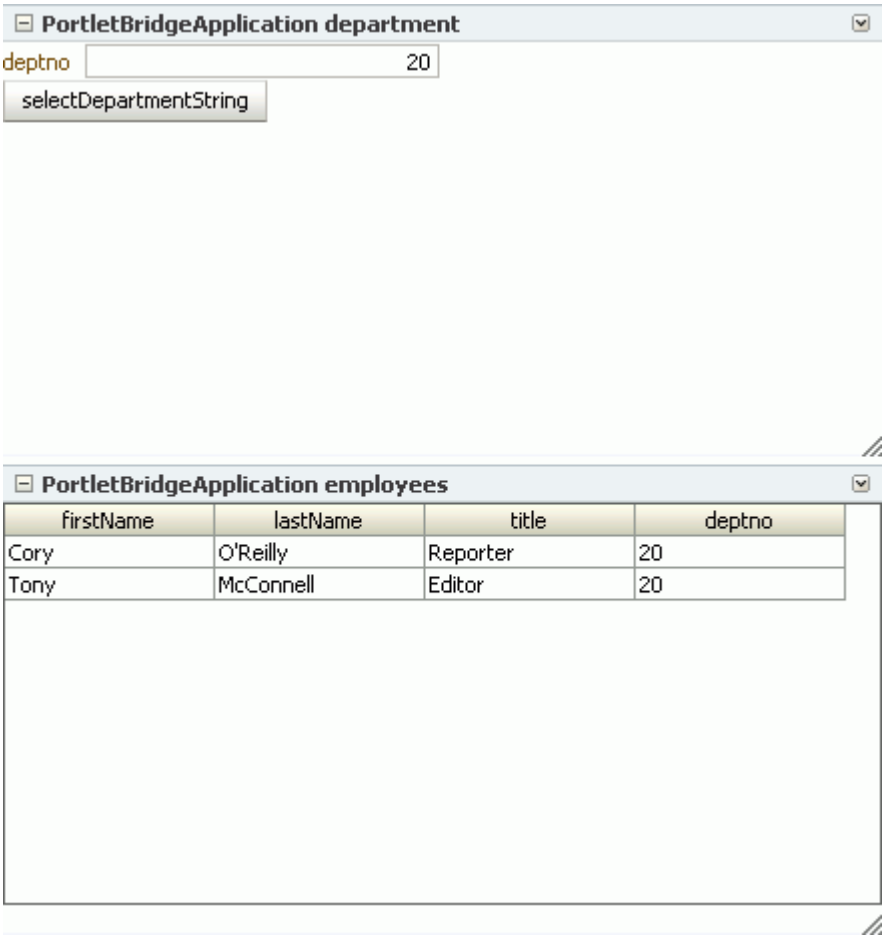
Who Is the Intended User?

Application developers with the knowledge of Faces and WSRP.

When Should They Be Used?

JSF portlets are best suited when application developers intend to display contents from a JSF application as a portlet without hosting the entire application, or without separately building a portlet for the same. When portletized, the consumption of the portlet is same as registering any WSRP producer using their provider URLs. ADF components that is task flows, can be exposed using the Oracle JSF Portlet Bridge as well.Figure 58–3 shows two portlets, one where users can enter a department number, and one that displays the employee information for the specified department. These portlets were created from ADF task flows using the Oracle JSF Portlet Bridge.

Figure 58–3 JSF Portlets



58.1.2.2 Prebuilt Portlets

What Are They?

Prebuilt portlets are available through Oracle's partnerships with leading system integrators, software vendors, and content providers. You can access these portlets by using the keywords `portal` or `portlet` when searching the Oracle PartnerNetwork Solutions Catalog, available at `http://solutions.oracle.com`. Examples of these include portlets for the following purposes:

- Generating point-to-point driving directions
- Accessing Information Technology (IT) information from a wide variety of sources
- Viewing summary information about news, stocks, and weather

Who Is the Intended User?

Fully developed, downloadable portlets are best suited for use by application developers who understand how to download, install, and register producers in WebCenter Portal: Framework. They are available for use by all levels of experience.

When Should They Be Used?

Use prebuilt portlets when your needs are satisfied by the functions the portlets offer and the level of personalization readily available is sufficient to complete the desired task.

Consider alternatives when you want to extend or personalize the portlet, for example, when you need a different user interface or when the functionality you require is not available out of the box.

58.1.2.3 Parameter Form and Parameter Display Portlets

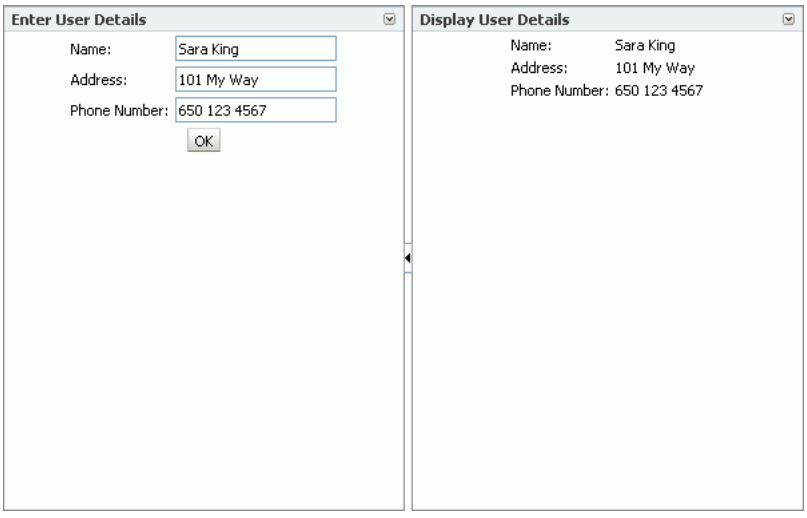
What Are They?

The Parameter Form and Parameter Display portlets provide a quick and easy way to pass values between components. They are provided by the WSRP Tools producer.

The Parameter Form portlet has three output parameters that are set when values are submitted in the form inside the portlet. The parameters can then be used to drive the content of other portlets. You can customize the Parameter Form portlet to determine how many of the three fields are displayed on the form, depending on how many parameters you require.

The Parameter Display portlet enables you to quickly test the wiring from the Parameter Form portlet. However, typically you use the values passed from the Parameter Form portlet to drive the content of some other portlet, for example to pass a zip code to a weather portlet, or a stock symbol to a stock ticker portlet.

Figure 58–4 Parameter Form and Display Portlets



Who Is the Intended User?

The Parameter Form and Parameter Display portlets are best suited for use by application developers who want to provide contextual linking between portlets on a page. These portlets can be added to a page by any user with the appropriate privileges.

When Should They Be Used?

Use the Parameter Form and Parameter Display portlets when your needs are satisfied by the functions the portlets offer and the level of personalization readily available is sufficient to complete the desired task.

Consider alternatives when you want to extend or personalize the portlet, for example, when you need a different user interface or when the functionality you require is not available out of the box.

58.1.2.4 Web Clipping

What Is It?

Web Clipping is a browser-based declarative tool that enables the integration of any web application with a Framework application. Web Clipping is designed to provide quick integration by leveraging the web application's existing user interface. Web Clipping has been implemented as a PDK-Java producer.

To create a Web Clipping portlet, the Framework application developer uses a web browser to navigate to a web page that contains desired content. Through the Web Clipping Studio, the application developer can drill down through a visual rendering of the target page to choose the desired content.

Web Clipping supports the following:

- **Navigation through various styles of login mechanisms,** including form- and JavaScript-based submission and HTTP Basic and Digest Authentication with cookie-based session management.
- **Fuzzy matching of clippings,** If a Web clipping gets reordered within the source page or if its character font, size, or style changes, then it is still identified correctly by the Web Clipping engine and delivered as the portlet content.
- **Reuse of a wide range of web content,** including basic support of pages written with HTML 4.0.1, JavaScript, applets, and plug-in enabled content, retrieved through HTTP GET and POST (form submission).
- **Personalization,** allowing an application developer to expose input parameters that users can modify when they personalize the portlet. These parameters can be exposed as public parameters that an application developer can map as page parameters. This feature enables users to obtain personalized clippings.
- **Integrated authenticated web content through Single Sign-On,** including integration with external applications, which enables you to leverage Oracle Single Sign-On and to clip content from authenticated external web sites.
- **Inline rendering,** enabling you to set up Web Clipping portlets to display links within the context of the portlet. As a result, when a user clicks a link in the Web Clipping portlet, the results display within the same portlet. You can use this feature with internal and external web sites.
- **Proxy Authentication,** including support for global proxy authentication and authentication for each user. You can specify the realm of the proxy server and whether all users automatically log in using a user name and password you provide, each user logs in using an individual user name and password, or all users log in using a specified user name and password.
- **Resource Tunneling** of images.
- **Open Transport API** for customizing authentication mechanisms to clipped sites.

Who Is the Intended User?

Web Clipping is best suited for use by application developers and component developers who want to leverage an existing web page for rapid portlet development. This portlet can be added to a page by any user with the appropriate privileges.

When Should It Be Used?

Use Web Clipping when you want to repurpose live content and functionality from an existing web page and expose it in your Framework application as a portlet. Consider alternatives to change the way information is presented in the clipped portlet. That is, you do not want to control the User Interface (UI) or application flow, and you are accessing web-based applications. For a greater level of control, use OmniPortlet's Web Page data source instead of Web Clipping. (For more information, see Section 58.1.2.5, "OmniPortlet.")

The following are some examples of when you can consider using the Web Clipping portlet:

- **Stock chart portlet.** You want to create a portlet that displays the stock market's daily performance chart from your financial advisor's web site. You could clip this information from an external web site, even if your company is using a proxy.
- **Web mail portlet.** Your users want to access their confidential web mail accounts through a portlet and to display their in-boxes in the portlet.

For more information about using Web Clipping, see Chapter 66, "Creating Content-Based Portlets with Web Clipping."

58.1.2.5 OmniPortlet

What Is It?

OmniPortlet is a declarative portlet-building tool that enables you to build portlets against a variety of data sources, including XML files, character-separated value files (CSV, for example, spreadsheets), web services, databases, and web pages. OmniPortlet users can also choose a prebuilt layout for the data. Prebuilt layouts include tabular, news, bullet, form, chart, or HTML. HTML layout enables OmniPortlet users to write their own HTML and inject the data into the HTML. Figure 58–5 shows a portlet created with OmniPortlet.

Figure 58–5 Product Information Portlet



Like Web Clipping, OmniPortlet supports proxy authentication, including support for global proxy authentication and authentication for each user. You can specify whether all users automatically log in using a user name and password you provide, each user logs in using an individual user name and password, or all users log in using a specified user name and password.

Who Is the Intended User?

Business users with a minimum knowledge of the URLs to their targeted data and limited programming skills may find OmniPortlet a valuable tool.

When Should It Be Used?

Use OmniPortlet when you want to build portlets rapidly against a variety of data sources with a variety of layouts. Consider alternatives when you want complete control of the design and functionality of the portlet.

The following are some examples of when you can consider using OmniPortlet:

- **RSS news feed portlet:** You want to create a portlet that displays live, scrolling news information to your users. The data comes from a Really Simple Syndication (RSS) news feed, such as Oracle Technology Network Headlines. You also want the portlet to contain hyperlinks to the news source.
- **Sales chart portlet:** You want to present up-to-date information about your company's sales results. You also want to display data in the form of a pie chart, and your company stores its sales information in a remote relational database.

For more information about OmniPortlet, see Chapter 65, "Creating Portlets with OmniPortlet."

58.1.2.6 Programmatic Portlets

What Are They?

Programmatic portlets are portlets that you write yourself, in Java, using either the standard Java Portlet Specification (JPS) or PDK-Java. WebCenter Portal: Framework provides two declarative wizards for simplifying the creation of standards-based JSR 286 portlets and Oracle PDK-Java portlets. These wizards assist in the construction of the framework within which you create the portlet. Each wizard may include easy steps for the following:

- Configuring general portlet properties
- Specifying names and search terms
- Setting allowable content types and mapping display modes
- Specifying user-customizable preferences
- Adding security roles
- Enabling default caching
- Adding initialization parameters
- Adding public render parameters
- Adding portlet events

Who Is the Intended User?

Use of the wizards is easy, but the creation of portlet logic is best performed by experienced and knowledgeable Java developers who are comfortable with the Java Portlet Specification or PDK-Java and who understand the configuration of producers.

When Should They Be Used?

Use programmatic portlets when you have very specialized business rules or logic or when you require personalized authentication, granular processing of dynamic results,

and complete user interface control. Additionally, use programmatic portlets when you want to satisfy any of the following conditions:

Consider using the programmatic approach when the out-of-the-box portlets do not address your needs.

The following list provides a couple of examples of when you can consider using programmatic portlets:

- **Photo Album portlet:** You want to create a a portlet that facilitates uploading, storing, and viewing user photos.
- **Shopping Cart portlet:** You want to create a portlet that facilitates the viewing and purchasing of, for example, company-branded items, such as mouse pads, pens, flash drives, tee shirts, and the like.

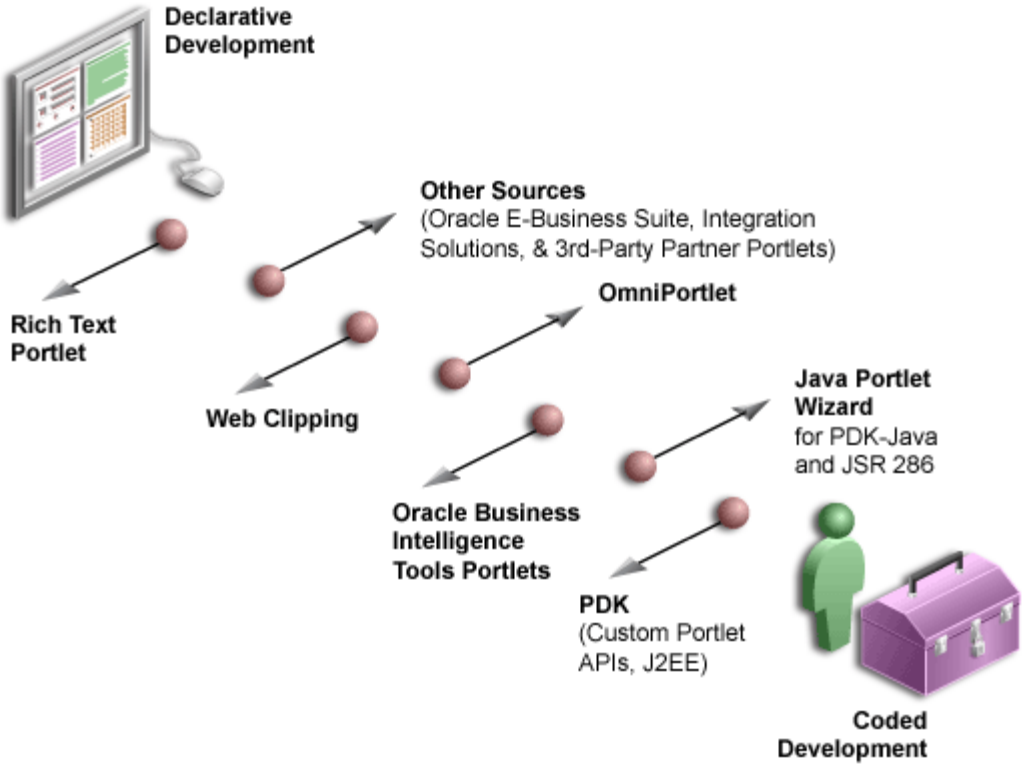
For more information about using programmatic portlets, see Chapter 60, "Creating Portlets with the Portlet Wizard" and Chapter 61, "Coding Portlets."

58.1.2.7 Deciding Which Tool to Use

Figure 58–6 illustrates the spectrum of portlet resources described in the previous section. Notice how one end of the spectrum is geared toward a more declarative development environment (that is, develop-through-wizard) while the other end focuses more on hand-coding. You can choose your tool depending on which type of environment is most comfortable and suitable to your skill-base.

For more information about deciding which tool to use, see Section 58.2, "Portlet Technologies Matrix."

Figure 58–6 Portlet Resources from Declarative to Coded Development



58.1.3 Portlet Modes

Portlet modes exhibit the runtime functionality seen by users. The standard modes supported by WebCenter Portal are described in the following subsections:

- Section 58.1.3.1, "View Mode"
- Section 58.1.3.2, "Edit Mode"
- Section 58.1.3.3, "Edit Defaults Mode"
- Section 58.1.3.4, "Help Mode"
- Section 58.1.3.5, "About Mode"

WebCenter Portal defines an extended set of modes that it supports. Different modes are available for both JSR 286 and PDK-Java portlets. For example, the Create JSR 286 Java Portlet wizard includes Print mode, which you can use to provide a printer friendly version of the portlet.

If you are coding portlets to JSR 286, then you can also declare your own custom portlet modes in the `portlet.xml` file. You can use these to map to the extra modes offered by PDK-Java (for example, Full Screen mode), or to accommodate any other functionality you may want to provide.

Defining custom modes is especially useful if the portlet must interoperate with portal implementations from other vendors. You can offer any of these modes to users. In fact, it is recommended that some modes like Edit Defaults are offered.

Note: Third party portal products may have their own set of extended modes (JSR 286 custom modes) that producers can offer. In Framework applications, the chrome UI for portlets only shows the custom modes that are defined in WebCenter Portal: Framework. Arbitrary custom modes that a third party or custom portlet producer offers are ignored and therefore are not supported.

58.1.3.1 View Mode

A portlet uses View mode (known as Shared Screen mode in PDK-Java) to appear on a page with other portlets. This is the mode most people think about when they envision a portlet. A JSR 286 portlet must have a View mode, the rest are optional.

For guidelines about how to implement View mode, see Section 61.1.1.1, "Guidelines for View Mode."

58.1.3.2 Edit Mode

A portlet uses Edit mode to enable users to personalize the behavior of the portlet. Personalization are visible only to the user that performed them; not to other users. Edit mode provides a list of settings that the user can change. These settings may include the title, type of content, formatting, amount of information, defaults for form elements, and anything that affects the appearance or content of the portlet.

Users typically access a portlet's Edit mode by choosing Personalize from the portlet's dropdown list of options. When users choose Personalize, a new page appears in the same browser window. The portlet typically creates a web page representing a dialog to choose the portlet's settings. After applying the settings, users automatically return to the original page.

For guidelines about how to implement Edit mode, see Section 61.1.1.2, "Guidelines for Edit Mode."

58.1.3.3 Edit Defaults Mode

A portlet uses Edit Defaults mode to enable administrators to customize the default behavior of a particular portlet instance. Edit Defaults mode provides a list of settings that the application developer can change. These settings may include the title, type of content, formatting, amount of information, defaults for form elements, and anything that affects the appearance or content of the portlet.

These default personalization settings can change the appearance and content of that individual portlet for all users. Because Edit Defaults mode defines the system-level defaults for what a portlet displays and how it displays it, this mode should not be used as an administrative tool or for managing other portlets.

Administrators access Edit Defaults mode, when editing a page, by choosing Customize from the portlet's dropdown list.

When users click the Customize icon, the portlet displays in the same browser window. The portlet typically creates a web page representing a dialog to personalize the portlet instance settings. After applying the settings, users are automatically returned to the original page.

For guidelines about how to implement Edit Defaults mode, see Section 61.1.1.3, "Guidelines for Edit Defaults Mode."

58.1.3.4 Help Mode

A portlet uses Help mode to display information about the functionality of the portlet and how to use it. The user should be able to find useful information about the portlet, its contents, and its capabilities with this mode.

Users access a portlet's Help mode by choosing the Help action in the portlet.

For guidelines about how to implement Help mode, see Section 61.1.1.4, "Guidelines for Help Mode."

58.1.3.5 About Mode

Users must be able to see what version of the portlet is currently running, its publication and copyright information, and how to contact the author. Portlets that require registration may link to web-based applications or contact information from this mode as well.

User's access a portlet's About mode by choosing About from the dropdown list in the portlet's chrome. A new page appears in the same browser window. The portlet can either generate the content for this new page or take the user to an existing page or application.

For guidelines about how to implement About mode, see Section 61.1.1.5, "Guidelines for About Mode."

58.2 Portlet Technologies Matrix

Table 58–1 summarizes the technologies and tools you can use with Oracle WebCenter Portal: Framework. The matrix describes the tools and technologies that are covered in more detail in this guide: Oracle JSF Portlet Bridge, OmniPortlet, Web Clipping portlet, and programmatic portlets, including standards-based (JSR 286) portlets and PDK-Java portlets.

Note: While these are the primary tools for building portlets, additional tools and technologies exist, such as other Oracle products, including Oracle Reports and Oracle BI Discoverer. These other tools are not covered in this guide.

Use Table 58–1 to quickly scan all the features and characteristics of the different portlet building technologies, then see the subsequent sections for more in-depth information.

Table 58–1 Portlet Building Technologies Comparison Matrix

JSF Portlets	Web Clipping	OmniPortlet	Programmatic Portlets Standards-Based/PDK-Java
General Suitability			
Oracle JSF Portlet Bridge exposes JSF and ADF artifacts as JSR 286 portlets.	A simple wizard-based tool, accessible from a browser, that assists in retrieving and presenting web content that originates from other web sites in a Framework application.	Wizard-based tool, accessible from a browser, that assists in retrieving and presenting data from a wide variety of data sources.	PDK-Java offers Oracle-specific application programming interfaces (APIs) for building portlets for use in Framework applications and Oracle Portal. Standards-based portlets additionally work with portals of other vendors. WebCenter Portal: Framework supports both WSRP and JSR 286 standards.
Expertise Required			
No expertise required.	No expertise required.	Basic understanding of one or more supported data sources and the concepts of portlet and page parameters.	Java, Servlet, JSP knowledge.
Supported Data Sources (for details, see Section 58.2.2, "Expertise Required")			
No limitations.	Any web site accessible on the network over HTTP or HTTPS.	CSV, XML, web service, SQL, web site, JCA.	No limitations.
Deployment Type			
WSRP producers using Oracle JDeveloper or manually.	PDK-Java producers	PDK-Java producers	PDK-Java uses PDK-Java producers. Standards-based portlets use WSRP producers.
Caching Style			
Expiry-based caching, validation-based caching (auto invalidate when personalized).	Expiry-based caching, validation-based caching (auto invalidate when personalized).	Expiry-based caching, validation-based caching (auto invalidate when personalized).	Expiry-based, validation, and invalidation caching, Edge Side Includes. **Note:** JSR 286 does not support validation based caching. WSRP 1.0 does. If you use a pure WSRP portlet, then validation-based caching is also supported. If you host a JSR portlet on WSRP (as is done in WebCenter Portal: Framework) then validation-based caching is not supported.
Development Tool			

Table 58–1 (Cont.) Portlet Building Technologies Comparison Matrix

JSF Portlets	Web Clipping	OmniPortlet	Programmatic Portlets Standards-Based/PDK-Java
Oracle JDeveloper Oracle JSF Portlet Bridge/JSF Portletization dialog.	Browser - wizard.	Browser - wizard.	Oracle JDeveloper Java Portlet wizard (or any other Java development environment).
Portlet Creation Style			
Develop a JSF application first, convert it into a JSF portlet later.	design time at runtime.	design time at runtime.	Develop first, add later.
User Interface Flexibility			
Very Flexible.	N/A	Very flexible, by using the HTML layout.	Very flexible.
Ability to Capture Content from Web Sites			
No. Depends upon the application being portletized.	Yes, by its nature.	Yes, by using the Web Page Data Source.	Yes. For PDK-Java, use the `oracle.portal.provider.v2.*` package. For standards-based portlets, use the `java.net` package.
Ability to Render Content Inline			
Yes.	Yes	No. However, inline rendering can be achieved through public portlet parameters.	Yes. For PDK-Java, use private portlet parameters. Standards-based portlets include servlets and JSPs, using the method `PortletContext.getRequestDispatcher()`.
Charting Capability			
No.	No	Yes, 2D-3D charts.	Yes, using BI Beans.
Public Portlet Parameter Support			
Yes	Yes	Yes	Yes
Private Portlet Parameter Support			
Yes	No	No	Yes
Ability to Hide and Show Portlets Based on User Privileges			
Yes.	No, though it is possible to apply security managers that are not exposed through the user interface (UI).	No, though it is possible to apply security managers that are not exposed through the UI.	Yes. For PDK-Java, by using the Security managers. For standards-based portlets, the Servlet security model is supported by using methods such as `PortletRequest.isUserInRole()` and `PortletRequest.getUserPrincipal()`.
Multilingual Support			
Yes.	N/A	Yes	Yes
Pagination Support			

Table 58–1 (Cont.) Portlet Building Technologies Comparison Matrix

JSF Portlets	Web Clipping	OmniPortlet	Programmatic Portlets Standards-Based/PDK-Java
Yes.	N/A	No	Yes, programmatically
Authenticating to External Applications			
Through custom user attributes.	External application integration supported.	Basic authentication support if the data source requires it.	For PDK-Java portlets, external application integration is supported. LDAP integration is supported when the portlet is running behind the same firewall as the LDAP server. For standards-based portlets, though not specifically supported, external application support is feasible through custom user attributes.) LDAP integration is supported.

The rest of this chapter includes the following sections that provide more in-depth detail about the different portlet building technologies:

- Section 58.2.1, "General Suitability"
- Section 58.2.2, "Expertise Required"
- Section 58.2.3, "Deployment Type"
- Section 58.2.4, "Caching Style"
- Section 58.2.5, "Development Tool"
- Section 58.2.6, "Portlet Creation Style"
- Section 58.2.7, "User Interface Flexibility"
- Section 58.2.8, "Ability to Capture Content from Web Sites"
- Section 58.2.9, "Ability to Render Content Inline"
- Section 58.2.10, "Charting Capability"
- Section 58.2.11, "Public Portlet Parameter Support"
- Section 58.2.12, "Private Portlet Parameter Support"
- Section 58.2.13, "Ability to Hide and Show Portlets Based on User Privileges"
- Section 58.2.14, "Multilingual Support"
- Section 58.2.15, "Pagination Support"
- Section 58.2.16, "Authenticating to External Applications"

58.2.1 General Suitability

This section describes each portlet-building technology in terms of its usage characteristics (for example, wizard-based or programmatic).

JSF Portlets

The Oracle JSF Portlet Bridge is a dialog-based tool that enables page developers to expose their existing JSF applications and task flows as JSR 286 portlets. Using the

Oracle JSF Portlet Bridge to create JSF portlets from JSF applications does not require a technical background.

Web Clipping

Web Clipping is a simple wizard-based tool that assists with retrieving and presenting web content that originates from other web sites in a Framework application. Web Clipping does not require a technical background.

Examples of portlets you can build using Web Clipping

The examples of portlets that you can build by using Web Clipping are as follows:

- Stock chart portlet
- Web mail portlet
- News portlet containing dynamic content from an existing web site

OmniPortlet

OmniPortlet is an easy-to-use, wizard-based tool for presenting information from a wide variety of data sources in a variety of formats. OmniPortlet runs completely in the browser. Drop OmniPortlet on a page, click the Define link, and choose a data source and a presentation format. Select from a wide variety of data sources as follows:

- Spreadsheet
- SQL
- XML
- Web Service
- Web page

OmniPortlet does not require the use of an additional development tool or a strong technical background. Even so, it can be used for building reusable, high-performing portlets.

Examples of portlets you can build using OmniPortlet

The examples of portlets that you can build by using OmniPortlet are as follows:

- RSS news feed portlet
- Sales chart portlet

Programmatic Portlets

If the wizard-based portlet building tools do not satisfy your needs, then you can build your portlets programmatically using Java. The Java Community Process standardized the Java portlet APIs in 2003. Portlets built against the Java Specification Request (JSR) 286 standard are interoperable across different portal platforms. The Java Portlet Wizard, a tool available through WebCenter Portal: Framework, assists with building Java portlets.

> **Note:** When building portlets in Java, you have full control over your portlet's functionality. For example, you can control what it looks like and how it behaves.

Examples of portlets you can build using Java

The examples of portlets that you can build by using Java are as follows:

- Discussion forum portlet
- E-mail portlet

58.2.2 Expertise Required

While some portlet building tools do not require portlet development skills, others assume a strong technical background. This section describes each tool in terms of the level of knowledge required to use it effectively.

JSF Portlets

The Oracle JSF Portlet Bridge does not require a technical background. However, you must have an understanding of Faces and WSRP.

Web Clipping

Web Clipping does not require a technical background. However, to parameterize the web page content that you clipped, you must have an understanding of public portlet parameters and page parameters.

OmniPortlet

OmniPortlet requires a basic knowledge of the data source you want to leverage in your portlet. Table 58–2 lists the types of data sources that can be used with OmniPortlet and describes the type of information required to work with each type.

Table 58–2 OmniPortlet Data Sources

Data Source	Required Information
Spreadsheet	The URL that points to the spreadsheet containing the data to display in the portlet.
SQL	The connection information to the data source and the SQL query that retrieves the data from the database.
XML	The location of the XML source and optionally the address of the XSL filter and the XML schema.
Web service	The Web Services Description Language (WSDL) URL, the method of the web service, and optionally the XSL filter URL and the XML schema URL.
Web page	The Web Page data source uses the same environment as Web Clipping. No technical background is required.
J2EE Connector Architecture	Although not displayed on the OmniPortlet Wizard's Type page, a J2EE Connector Architecture (JCA) 1.0 adapter is also available. JCA provides a mechanism to store and retrieve enterprise data such as that held in ERP systems (Oracle Financials, SAP, PeopleSoft, and so on).

Programmatic Portlets

To build Java portlets, you must know at least a subset of Java EE. Knowing HTML, Java servlets, and XML is a must, and JSP experience is recommended. Additional Java knowledge is optional, depending on the task you want to perform. Using Java portlets you can access any data source supported by the Java language.

58.2.3 Deployment Type

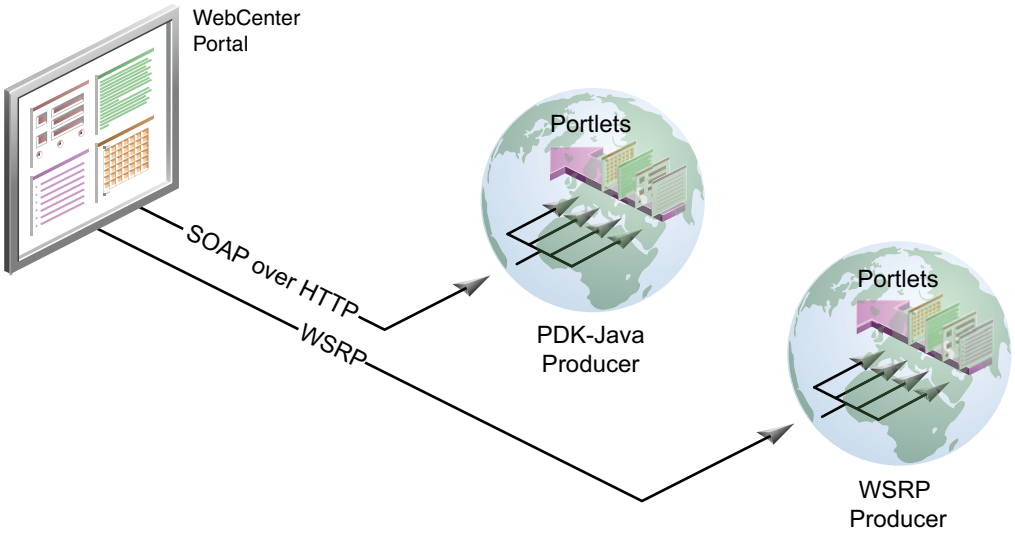
Before a portlet can be consumed by an application, you must first deploy it, then register the producer you've deployed the portlet to. As shown in Figure 58–7, portlets can be deployed through the following two producer types:

- PDK-Java producers
- WSRP producers

Within WSRP, both version 1.0 and 2.0 are supported.

PDK-Java portlets are deployed to a Java EE application server, which is often remote and communicates with the consumer through Simple Object Access Protocol (SOAP) over HTTP. JSR 286 portlets are deployed to a WSRP producer, which is also remote and communicates with the consumer through WSRP (Web Services for Remote Portlets). JSF Portlets are deployed as JSR 286 portlets, that is, to a WSRP producer, either from Oracle JDeveloper or manually.

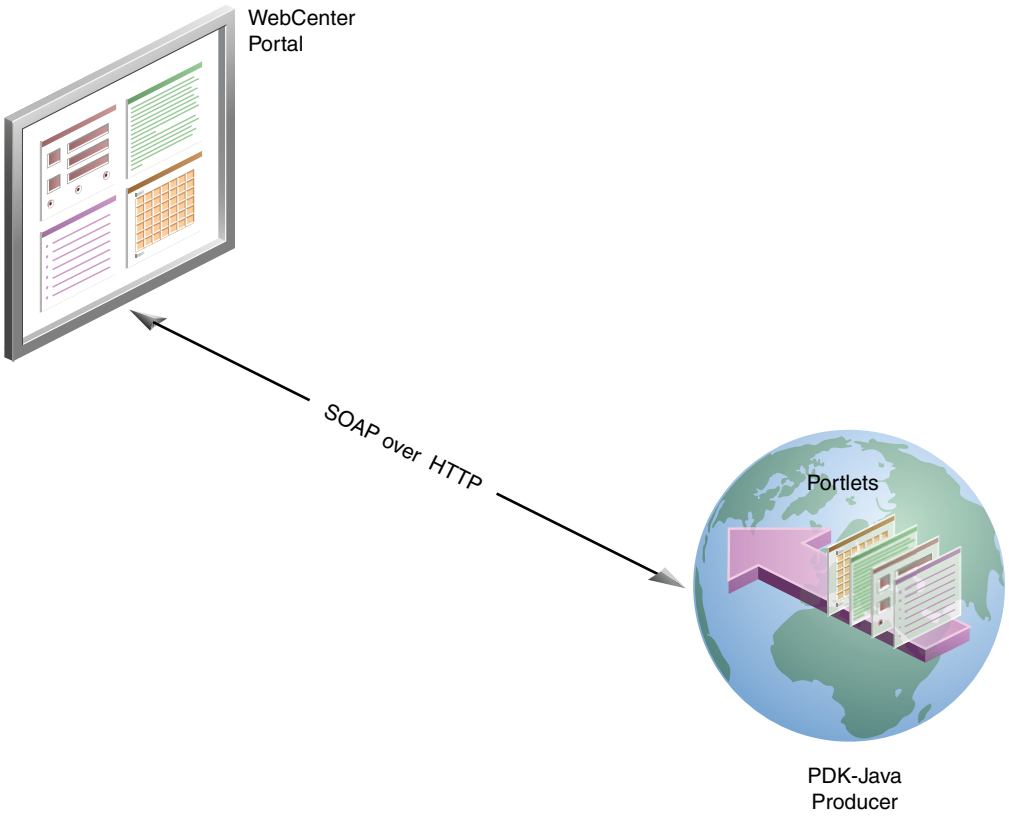
Figure 58–7 Portlet Producer Overview



58.2.3.1 PDK-Java Producers

PDK-Java producers use open standards, such as XML, SOAP, HTTP, or Java EE for deployment, definition, and communication with applications. Figure 58–8 shows how Oracle Portal incorporates portlets from a PDK-Java producer and the PDK-Java producer communicates with Framework application using SOAP over HTTP.

Figure 58–8 PDK-Java Producers



There are several benefits to developing portlets and exposing them through PDK-Java producers as follows:

- Deploy portlets remotely.
- Leverage existing web application code to create portlets.
- Specify producers declaratively.
- Use standard Java technologies (for example, servlets and JSPs) to develop portlets.

To expose your portlets using a PDK-Java producer, you must first create a producer that manages your portlets and communicates with WebCenter Portal: Framework using SOAP. To learn how to expose your portlets using a PDK-Java producer, see Section 60.2, "Creating Java Portlets."

58.2.3.2 WSRP Producers

WebCenter Portal supports Web Services for Remote Portlets (WSRP) versions 1.0 and 2.0. This emerging standard provides support for inter-portlet communication and export or import of portlet customizations.

The Producer Registration wizard is the entry point for registering both WSRP 1.0 and 2.0 producers. The wizard automatically recognizes whether WSRP 1.0 or 2.0 is in play.

Architecturally, WSRP producers are very similar to PDK-Java producers. WSRP is a communication protocol between Framework application servers and portlet containers. WSRP is a standard that enables the plug-and-play of visual, user-facing web services with intermediary web applications.

Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as JSR 286, .NET, Perl) and any WSRP portal. So, a portlet (regardless of language) deployed to a WSRP-enabled container can be rendered on any application that supports this standard.

> **Note:** For more information about the WSRP architecture, see "The Relationship Between WSRP and JSR 286" in Chapter 60, "Creating Portlets with the Portlet Wizard."

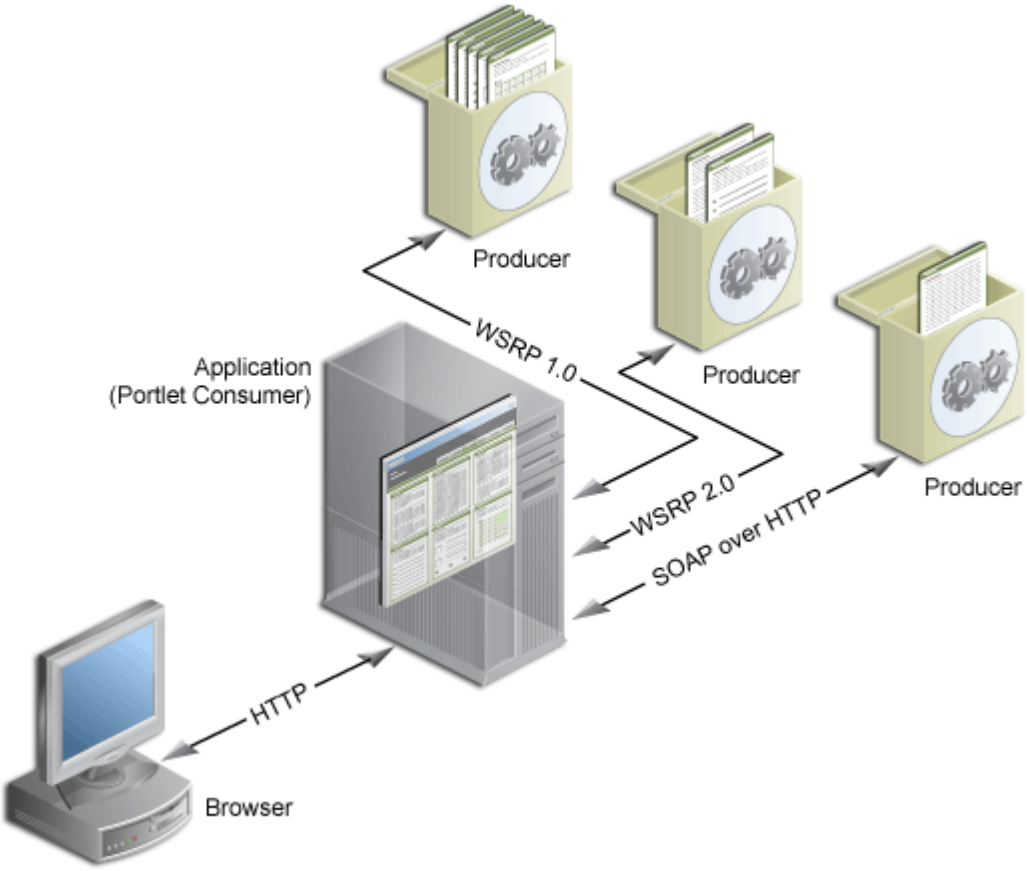
To make standard portlets (such as JSR 286, .NET, Perl) available to a Framework application, you must package them in a Portlet Producer application and deploy them to a WSRP container.

To learn how to package your portlets in a WSRP container, see Chapter 62, "Testing and Deploying Your Portlets."

58.2.3.3 Producer Architecture

Figure 58–9 illustrates the basic architecture of portlet producers.

Figure 58–9 Producer Architecture



When users display a page in their web browsers, the flow of the request works as follows:

1. The user requests a page from the web browser by entering a URL in the browser's address field.

2. The browser transmits the request to the application over HTTP.
3. The application contacts the portlet producers that provide the portlets that display on the requested page.
4. The producers make the necessary calls to their portlets so that the portlets generate the portlet content in the form of HTML or XML code.
5. The producers return the portlet content back to the application using their relevant protocols:
   - JSR 286 portlets and JSF portlets are initialized by WSRP producers, which communicate using the WSRP 1.0 or 2.0 protocol.
   - PDK-Java portlets are initialized by PDK-Java producers, which communicate using SOAP over HTTP.

JSF portlets, Web Clipping, OmniPortlet, and Java portlets communicate with WebCenter Portal: Framework through either WSRP or PDK-Java producers. You must register these producers with WebCenter Portal: Framework before you can use the portlets they produce in your Framework application.

The latest versions of Web Clipping and OmniPortlet are available through Application Development Runtime Service (ADRS). For more information, see Section 3.4, "Working with the Integrated WebLogic Server."

58.2.4 Caching Style

Portlet caching is key to rapid response to user requests. Portlets implement validation-based and expires-based caching using Java Object Cache. Invalidation-based caching continues to be implemented by a web cache that fronts the PDK-Java producer running the Web Clipping portlet and OmniPortlet.

Caching rules can be specified at a portlet's container level, encoded in the portlet's own logic, or, for JSR 286 portlets, established through the portlet wizard. Provided it is specified, container-level caching takes over when caching is not part of the portlet code.

At the application level, WebCenter Portal: Framework supports use of a Java cache for the establishment of application-level caching rules.

When not using caching, you may find accessing various data sources with Web Clipping and OmniPortlet to be time consuming. When you enable caching at the application level, you instruct the Java cache to maintain a copy of the portlet content. When data that was previously cached is requested, no time is lost in contacting the data source and regenerating its content. Instead, the previously cached portlet content is returned.

A portlet's content weighs heavily in determining the type of caching the portlet should use. For example:

- **Expiry-based caching**: Consider using expiry-based caching when the portlet content is static or when it is not critical that the most up-to-date content be displayed. When using expiry-based caching, you must specify the caching period.
- **Validation-based caching:** Consider using validation-based caching for portlets with dynamic content that changes frequently or unpredictably. The portlet associates its content with a caching key and returns the key value along with the content. When the portlet content is requested, the portlet decides, based on the caching key, if the current content is valid. If the portlet content is valid, then it returns a response indicating that the cached content can be used (that is, the

content is valid) or generates the new portlet content and returns it along with a new caching key for that content.

JSF Portlets

Oracle JSF Portlet Bridge supports expiry- and validation-based caching. JSF portlets can be cached in full, or you can use Edge Side Includes (ESI) to cache fragments of portlets.

Web Clipping and OmniPortlet

In addition to invalidation-based caching, expiry-based caching can be specified for the Web Clipping portlet and OmniPortlet. Additionally, these portlets are refreshed automatically when they are personalized.

Programmatic Portlets

Java portlets support expiry- and validation-based caching. These portlets can be cached in full, or you can use Edge Side Includes (ESI) to cache fragments of portlets.

58.2.5 Development Tool

This section describes the development tools you can use to build different types of portlets.

JSF Portlets

Oracle JSF Portlet Bridge/JSF Portletization dialog is used to create a JSF portlet based upon a page or a task flow.

Web Clipping and OmniPortlet

OmniPortlet and Web Clipping use a browser-based wizard as the development tool.

Programmatic Portlets

Although you can use any Java development environment to build Java portlets, it is highly recommended that you use Oracle JDeveloper, a professional IDE. While you can consider other IDEs, Oracle JDeveloper includes the Java Portlet Wizard, to minimize your Java portlet development efforts.

The Java Portlet Wizard generates a starting skeleton and file structure for both JSR 286 and PDK-Java portlets. You need only add your own business logic to the skeleton. Oracle JDeveloper can also package and deploy your applications to your Java EE container. Also, Oracle JDeveloper helps you test your portlet producer. Oracle recommends that you use the Integrated WLS, provided through WebCenter Portal: Framework, as your development Java portlet runtime environment. For more information, see Chapter 3, "Preparing Your Development Environment."

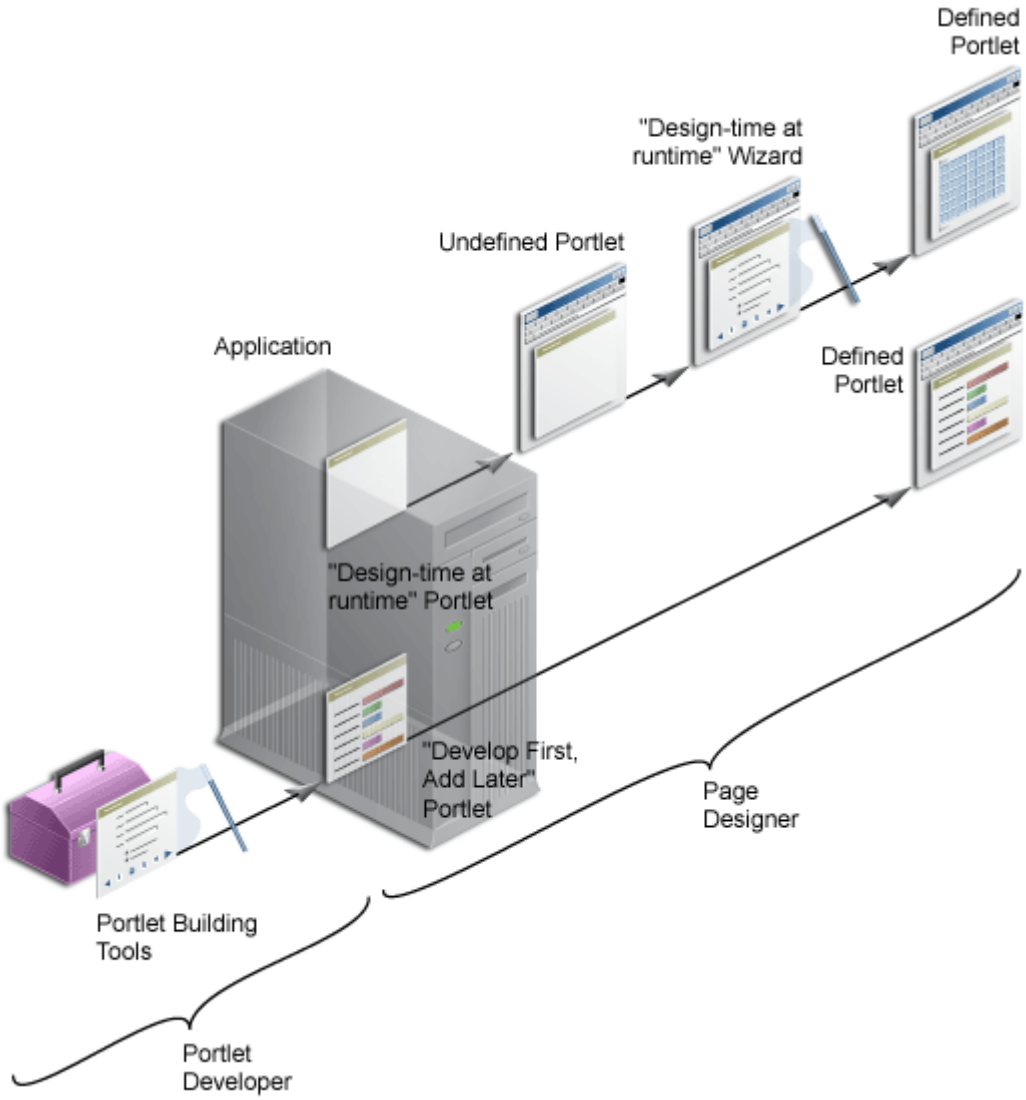
58.2.6 Portlet Creation Style

WebCenter Portal: Framework supports the following types of portlet creation (Figure 58–10):

- Develop first, add later
- Design time at runtime

Develop first, add later portlet creation is usually the task of the *portlet* developer; design time at runtime portlet creation is the *application* developer's responsibility.

Figure 58–10 Portlet Creation Style



JSF Portlets

Develop a JSF application first and then portletize the application using the JSF Portletization dialog at design time.

OmniPortlet and Web Clipping

OmniPortlet and Web Clipping both offer a "design time at runtime" portlet creation style. Register the portlet producers with the application that consumes the portlets, add the portlets to an application page, run the application, and then define the portlets in-place on the page.

Programmatic Portlets

Typically programmatic portlets offer a "develop first, add later" portlet creation style. Two wizards are available through WebCenter Portal: Framework to assist with the creation of Oracle PDK-Java and JSR 286 portlets. The wizards generate the basic files required for portlet creation. The developer hand-codes the portlet logic. The development sequence for programmatic portlets is to create the portlet, deploy it to a

producer, register the producer with the application that consumes the portlet, and then add the portlet to an application page.

Note: With extensive coding, you can create "design time at runtime" Java portlets. For example, Web Clipping and OmniPortlet are both Java portlets.

58.2.7 User Interface Flexibility

This section describes the portlet building tools in terms of the control you have over the user interface.

JSF Portlets

Oracle JSF Portlet Bridge ensures that the portletized JSF application functions as a JSF portlet.

Web Clipping

Because of its nature, Web Clipping always displays the remote web site content, therefore UI flexibility is not a requirement for this portlet.

OmniPortlet

OmniPortlet enables you to use different pre-built layouts, such as scrolling news, tabular, and chart. You can also use the built-in HTML layout to personalize the look and feel of your portlet using HTML and JavaScript.

Note: When using JavaScript in portlets, developers must ensure that the JavaScript identifiers are qualified. That is, identifiers must be unique for each portlet instance and must not clash with the JavaScript on the page.

Programmatic Portlets

In Java portlets, you have full control over your portlet's user interface. Your portlet is free to generate any HTML content that conforms to the rendering rules for pages.

58.2.8 Ability to Capture Content from Web Sites

This section describes the portlet building tools in terms of their ability to include content from other sources.

JSF Portlets

Oracle JSF Portlet Bridge's capability to capture contents from web sites depends upon the JSF application being portletized.

Web Clipping

For portlets that display content from a remote web site as it is presented at the source location, the best tool to use is Web Clipping. Web Clipping can tolerate the changes of the source HTML page to some extent. If a clipped table moves from one place to another in the source page, then the Web Clipping engine can find the table again using the internal "fuzzy match" algorithm. Portlets built with Web Clipping can also maintain sessions to the remote web sites. Web Clipping also supports user personalization of HTML form values.

OmniPortlet

For portlets using the data but not the layout from a remote web site, the best choice is OmniPortlet. Use OmniPortlet to retrieve the data, process the data (format, filter, and so on), and present it in a portlet in a tabular, chart, or news format. OmniPortlet is a powerful tool that extracts data from web pages by using its Web Page data source.

Programmatic Portlets

Java portlets can take advantage of low-level Java networking APIs to retrieve and process content from remote web sites. To avoid unnecessary development efforts, before choosing Java always ensure that Web Clipping or OmniPortlet are not viable options.

58.2.9 Ability to Render Content Inline

Active elements in portlets, such as links or form buttons, enable users to navigate to remote URLs. In a News portlet, for example, a user can click a hyperlink to navigate to a news site with detailed information about news of interest. For example, a user clicks a news-summary link in a News portlet, leaves the application page, and lands on the news site.

You may have a requirement to keep your users within the context of the application page by rendering the requested content within the same portlet container. For example, a user clicks a news-summary link in a News portlet, and the portlet refreshes with the detailed news article.

This maintenance of context is what rendering content inline means .

JSF Portlets

Oracle JSF Portlet Bridge converts a JSF application with multiple pages into JSF portlets. The contents of these pages are rendered inline.

Web Clipping

The Web Clipping portlet supports URL rewriting for achieving inline content rendering. It can process the links originating from the source web site and rewrite them to achieve the desired functionality.

The following options are available:

- Select not to rewrite the URLs within the portlet, in which case clicking the links takes users out of the Framework application to the web site that provides the clipping. Whenever the link brings the user to a place that requires authentication, the user must enter login information before the link target is displayed.
- If the Web Clipping provider is registered with an external application and the clipping requires authentication, then you can instruct Web Clipping to rewrite all URLs within the portlet to point to the Login Server. In this case, navigation causes the user to leave the Framework application, while also using the Login Server to log the browser into the External Application.
- Select to rewrite all URLs within the portlet (inline rendering) to point back to the page so that all browsing within the Web Clipping portlet remains within the Framework application. If the Web Clipping provider is registered with an External Application, then this causes the Web Clipping provider to log itself in to the External Application. In this case, the navigation within the Framework application through the Web Clipping provider is authenticated in the External Application.

OmniPortlet

Rendering content inline is not supported, but you can achieve inline rendering using public portlet parameters.

Programmatic Portlets

As you have full control over the links and buttons in Java portlets, you can easily implement the inline rendering functionality. To achieve inline rendering, you must append the private portlet parameters to the page URL.

If you use Struts in your portlet, then the PDK-Struts integration framework renders your content always in the same portlet container. Oracle recommends, however, that you use ADF Faces navigation for your new WebCenter Portal: Framework portlets.

If your portlet consists of multiple JSPs (for example, several steps in a survey or wizard), then your portlet can make use of a special parameter to specify at runtime the JSP to use to render the content.

58.2.10 Charting Capability

This section describes the portlet building tools in terms of their charting capability.

JSF Portlets

Oracle JSF Portlet Bridge does not have charting capability. However, it supports portletization of a JSF application that contains ADF charts.

Web Clipping

Web Clipping clips pre-existing content. So, while it does not create charts, it can retrieve and present HTML content that contains charts.

OmniPortlet

OmniPortlet supports bar, line, and pie chart types. Charts are dynamically generated images, which can include hyperlinks.

Programmatic Portlets

You can create sophisticated charts programmatically in Java portlets using Oracle's Business Intelligence (BI) Beans.

Note: Oracle Reports and Oracle Discoverer portlets use BI Beans to create professional graphs.

58.2.11 Public Portlet Parameter Support

Typically, a portlet's state is opaque (private); however, in WebCenter Portal: Framework portlets can describe public inputs (parameters) so values can be coordinated by the consuming application with other constituents of that application.

Inputs can include, public portlet parameters and private portlet parameters. These can be described as follows:

- **Public portlet parameters:** Use public portlet parameters to pass values to a portlet. Public parameters can assist with rendering portlet content that is specific to a particular page or user. Portlet parameters are created by the component developer and then exposed to the application developer through the user interface. After adding a portlet to a page, application developers can assign

values to public portlet parameters to make the information displayed in the portlet specific to the page.

Assigned values can be specific (such as a constant), a system variable (for example, the user name), or a page parameter. At runtime, the portlet receives the values from the sources specified.

- **Private portlet parameters:** Use private portlet parameters to implement internal navigation in a portlet. Parameters are passed to the portlet every time the page is requested. Private portlet parameters can be passed exclusively from the portlet instance to the same portlet instance. Private portlet parameters do not require a full page refresh.

  Note: A Refresh action is available for inclusion on the portlet's Actions menu (`isNormalModeAvailable`). It refreshes the portlet without triggering a full page refresh. For details about setting this action programmatically, see Section 64.4, "Setting Attribute Values for the Portlet Tag."

Portlets supporting public portlet parameters enable application developers to tailor data input for each portlet instance. The component developer can focus on the portlet logic, while the application developer can address the interaction between the application page and its portlets.

All portlet building technologies discussed in this chapter (OmniPortlet, Web Clipping, and programmatic portlets) support public portlet parameters. OmniPortlet and Web Clipping provide complete support through their wizard interface. You can add public portlet parameter support to your programmatic portlets programmatically or with the Java Portlet wizard.

58.2.12 Private Portlet Parameter Support

This section describes the portlet building tools in terms of their support for private parameters.

JSF Portlets

In your JSF portlets, you can implement internal navigation using private portlet parameters.

OmniPortlet and Web Clipping

With the OmniPortlet and Web Clipping portlets, component developers do not have access to private portlet parameters.

Programmatic Portlets

In your Java portlets, you can implement internal navigation using private portlet parameters.

58.2.13 Ability to Hide and Show Portlets Based on User Privileges

This section describes the portlet building tools in terms of their support for authorization functionality.

JSF Portlets

Oracle JSF Portlet Bridge supports the standard servlet mechanisms.

Web Clipping and OmniPortlet

Dynamically hide and show portlets built with Web Clipping and OmniPortlet by using security managers. Although Web Clipping and OmniPortlet do not expose security managers through the user interface, they make them available for editing through their XML provider definition files.

Programmatic Portlets

PDK-Java provides security managers for Java portlets. For example:

- **Group security manager:** The group security manager controls access to portlets based on group membership. For example, it shows the portlet to users who are members of a specified group, and hides the portlet from non-members.
- **Authentication level security manager:** The authentication level security manager controls access to the portlets based on authentication level. For example, it shows the portlet to authenticated users, and hides it from public users.

JSR 286 portlets support the standard servlet mechanisms.

58.2.14 Multilingual Support

This section describes the portlet building tools in terms of their support for other languages.

Web Clipping, OmniPortlet, JSF portlets, and Java portlets display textual information in the language selected by the end user.

58.2.15 Pagination Support

Support for pagination is useful when a portlet must display a relatively large set of records.

JSF Portlets

Oracle JSF Portlet Bridge supports pagination, if the application portletized has pagination implemented in it.

Web Clipping

Web Clipping does not support pagination.

OmniPortlet

OmniPortlet does not support pagination.

Programmatic Portlets

With Java, portlet pagination can be implemented programmatically.

58.2.16 Authenticating to External Applications

This section describes the portlet building tools in terms of authentication for external applications.

JSF Portlets

Since any portletized application can function as a WSRP portlet, the external application support is feasible through custom user attributes.

Web Clipping

Web Clipping's integration with the external application framework provides a fully automated mechanism to store passwords to external web sites. All you must do is provide an External Application ID when registering the Web Clipping producer.

OmniPortlet

OmniPortlet enables you to store connection information when the data source is password protected. The credentials to access the data source can either be shared across all users or saved individually for each user. OmniPortlet can storing database credentials and HTTP basic authentication user name-password pairs. Credentials are stored in a secured metadata services repository.

Programmatic Portlets

Java portlets support programmatic integration with the external application framework and any LDAP server, such as Oracle Internet Directory.

59

Creating Portlets with the Oracle JSF Portlet Bridge

This chapter explains how to use the Oracle JSF Portlet Bridge to expose an application as a portlet.

This chapter includes the following sections:

- Section 59.1, "Introduction to the Oracle JSF Portlet Bridge"
- Section 59.2, "Creating a Portlet from a JSF Application"
- Section 59.3, "What You May Need to Know When Creating a JSF Portlet"
- Section 59.4, "Copying a Runtime-Created Skin to a JSF Portlet Producer Application"

59.1 Introduction to the Oracle JSF Portlet Bridge

The Oracle JSF Portlet Bridge enables application developers to quickly and easily expose their existing JSF applications, and Oracle ADF applications and task flows as JSR 286 portlets.

> **Note:** Unless otherwise noted, the term JSF applications encompasses Oracle ADF applications as well.

The Oracle JSF Portlet Bridge:

- Simplifies portlet development by enabling you to provide portlet functionality using JSF rather than relying on the JSR 286 portlet APIs.
- Simplifies exposing your JSF application to JSR 286 portlet consumers, such as Oracle Portal and Oracle WebCenter Portal: Spaces.
- Eliminates the requirement to store, maintain, and deploy your portlets separately from your application by enabling the application to run simultaneously as a regular web application and as a portlet from the same installation.
- Enables you to create portlets at a more granular level by exposing task flows as portlets. Because portletized task flows are WSRP portlets, this also enables you to use distributed task flows.

Note: The Oracle JSF Portlet Bridge is based on and conforms to JSR 329. JSR 329 is the standards effort to define the functionality for the Portlet 2.0 Bridge for JavaServer Faces. Oracle is the specification lead for this standard. More information is available at:

`http://www.jcp.org/en/jsr/detail?id=329`

59.2 Creating a Portlet from a JSF Application

The Oracle JSF Portlet Bridge enables you to expose a JSF application or task flow as a portlet. You do this declaratively, using the Create Portlet Entry dialog; no coding is required. Using the Create Portlet Entry dialog, you can configure Oracle JSF Portlet Bridge on a JSF application to expose the application as a JSR 286 portlet. As part of this configuration, you indicate the initial JSF view (or task flow view) to invoke when the portlet is rendered. From that point on the Oracle JSF Portlet Bridge works with the JSF application to navigate through the additional views that are reachable from this initial view. So in the typical situation when you are exposing the entire JSF application as the portlet, you configure the Oracle JSF Portlet Bridge to render the application's initial view in the portlet and the rest of the navigation works naturally within that same portlet.

This section includes the following subsections:

- Section 59.2.1, "How to Create a JSF Portlet Based on a Page"
- Section 59.2.2, "How to Create a JSF Portlet Based on a Task Flow"
- Section 59.2.3, "How to Test a JSF Portlet Using the Integrated WebLogic Server"
- Section 59.2.4, "What Happens at Runtime"

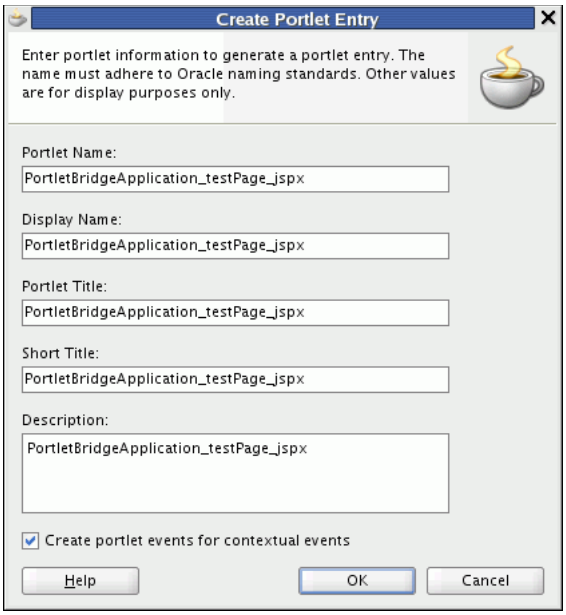
59.2.1 How to Create a JSF Portlet Based on a Page

The simplest way to create a portlet from a JSF application is to generate a portlet based upon a page.

To create a JSF portlet from an existing application page:

1. In the JDeveloper Application Navigator, open the application that contains the `.jspx` page to portletize.
2. Right-click the page to portletize and choose **Create Portlet Entry.**
3. In the Create Portlet Entry dialog (Figure 59–1), in the **Portlet Name** field, enter a name for the portlet.

Figure 59–1 The Create Portlet Entry Dialog for a Page



4. In the **Display Name** field, enter a descriptive name for your portlet.
5. In the **Portlet Title** field, enter a descriptive title for your portlet.

   The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.
6. In the **Short Title** field, enter a shorter title for your portlet. This short title is displayed on the portlet header when the portlet appears on a page on a mobile device.
7. In the **Description** field, enter a description for your portlet.
8. Select **Create portlet events for contextual events** to create portlet events in the `portlet.xml` file for any contextual events exposed by the page. This option is selected by default.

   Portlet events enable a portlet to communicate with the page on which it resides and with other portlets on that page.
9. Click **OK.**

   The `portlet.xml` file is created.

   This file contains the portlet entry (Example 59–1) and is opened ready for viewing or editing. By default, the file is opened in Design view. To view or edit the source code, click the **Source** tab.

Example 59–1 Generated Portlet Entry for a Page

```
<portlet id="adf_jsf__testPage_jspx">
  <description>PortletBridgeApplication_testPage_jspx</description>
  <portlet-name>PortletBridgeApplication_testPage_jspx</portlet-name>
  <display-name>PortletBridgeApplication_testPage_jspx</display-name>
  <portlet-class>
    oracle.portlet.bridge.adf.application.ADFBridgePortlet
  </portlet-class>
```

```
  <init-param>
    <name>javax.portlet.faces.defaultViewId.view</name>
    <value>/myPage.jspx</value>
  </init-param>
  <supports>
    <mime-type>text/html</mime-type>
    <portlet-mode>VIEW</portlet-mode>
  </supports>
  <supported-locale>en</supported-locale>
  <portlet-info>
    <title>PortletBridgeApplication_testPage_jspx</title>
    <short-title>PortletBridgeApplication_testPage_jspx</short-title>
  </portlet-info>
  <supported-processing-event id="DepartmentSelectedEvent">
    <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
      x:DepartmentSelectedEvent
    </qname>
  <supported-publishing-event id="DepartmentSelectedEvent">
    <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
      x:DepartmentSelectedEvent
    </qname>
  </supported-publishing-event>
  <supported-public-render-parameter>
    _adf_event_DepartmentSelectedEvent
  </supported-public-render-parameter>
  <container-runtime-option>
    <name>com.oracle.portlet.requireIFrame</name>
    <value>true</value>
  </container-runtime-option>
  <container-runtime-option>
    <name>com.oracle.portlet.minimumWsrpVersion</name>
    <value>2</value>
  </container-runtime-option>
</portlet>
```

The page you selected earlier is used as the entry point for the portlet View mode. This is indicated in the `portlet.xml` file by the `javax.portlet.faces.defaultViewId.view` initialization parameter. You can manually edit the `portlet.xml` file to define the pages for other default and extended portlet modes supported by WebCenter Portal:

- Edit mode: `javax.portlet.faces.defaultViewId.edit`
- Help mode: `javax.portlet.faces.defaultViewId.help`
- About mode: `javax.portlet.faces.defaultViewId.about`
- Config mode: `javax.portlet.faces.defaultViewId.config`
- Edit Defaults mode: `javax.portlet.faces.defaultViewId.edit_defaults`
- Preview mode: `javax.portlet.faces.defaultViewId.preview`
- Print mode: `javax.portlet.faces.defaultViewId.print`

Note: The value for the `defaultViewId` is relative to the application context root and must always start with a `/`. For example, in Example 59–1, the value for `defaultViewId.view` is `/myPage.jspx`.

If you add `defaultViewId` for other portlet modes, then ensure that you also add the mode to the `<supports>` tag. For example, `<portlet-mode>HELP</portlet-mode>`.

59.2.2 How to Create a JSF Portlet Based on a Task Flow

An advantage of using Oracle ADF is task flows that provide a modular approach for defining control flow in an application. Instead of representing an application as a single large JSF page flow, you can break it up into a collection of reusable task flows. In each task flow, you identify application activities, the work units that must be performed in order for the application to complete. An activity represents a piece of work that can be performed when running the task flow.

Task flows can be unbounded or bounded:

- An unbounded task flow is a set of activities, control flow rules, and managed beans interacting to allow a user to complete a task. An unbounded task flow consists of all activities and control flows in an application that are not included within any bounded task flow.
- A bounded task flow is a specialized form of task flow, having a single entry point and one or more exit points. It contains its own set of private control flow rules, activities, and managed beans. An Oracle ADF bounded task flow allows reuse, parameters, transaction management, and reentry. It can have zero to many exit points.

A typical application is a combination of an unbounded and one or more bounded task flows. The application can then call bounded task flows from activities within the unbounded task flow. For more detailed information about bounded and unbounded task flows, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

This section includes the following subsections:

- Section 59.2.2.1, "Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog"
- Section 59.2.2.2, "Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog"

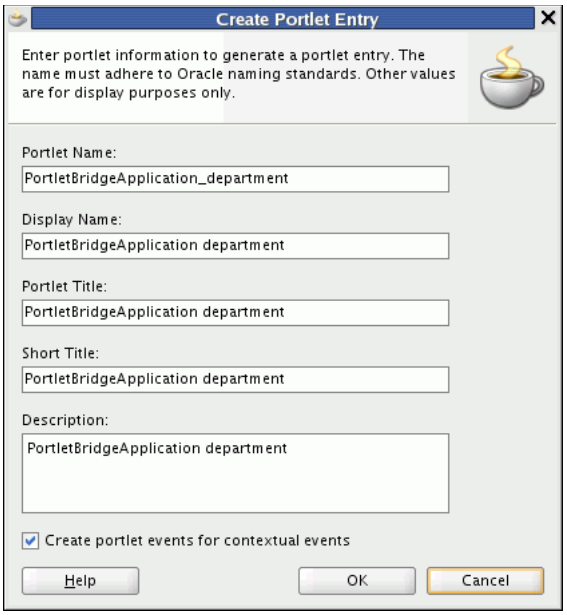
59.2.2.1 Creating a Portlet From a Task Flow Using the Create Portlet Entry Dialog

Use the Create Portlet Entry dialog to create a portlet from a single task flow.

To make a portlet from a task flow using the Create Portlet Entry dialog:

1. In the JDeveloper Application Navigator, open the JSF application that contains the task flow from which you want to make a portlet.
2. Right-click the task flow to portletize and choose **Create Portlet Entry.**
3. In the Create Portlet Entry dialog (Figure 59–2), in the **Portlet Name** field, enter a name for the portlet.

Figure 59–2 The Create Portlet Entry Dialog for a Task Flow



4. If the task flow is unbounded, the **Entry Point View** dropdown list displays all the view activities of the task flow. From this dropdown list, choose the view activity to use as the entry point for the portlet. By default, the first view activity in the task flow is chosen.

   If the task flow is bounded, there is only a single possible entry point, so you do not see the **Entry Point View** dropdown list.

5. In the **Display Name** field, enter a descriptive name for your portlet.

6. In the **Portlet Title** field, enter a descriptive title for your portlet.

   The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.

7. In the **Short Title** field, enter a shorter title for your portlet. This short title is displayed on the portlet header when the portlet appears on a page on a mobile device.

8. In the **Description** field, enter a description for your portlet.

9. Select **Create portlet events for contextual events** to create portlet events in the `portlet.xml` file for any contextual events exposed by the task flow. This option is selected by default.

   Portlet events enable a portlet to communicate with the page on which it resides and with other portlets on that page.

10. Click **OK.**

    When your portlet has been created, you should receive a message that says:

    ```
    New portlet has been successfully created
    ```

    In addition, the `portlet.xml` file is created. The `portlet.xml` file contains the portlet entry (Example 59–2) and is opened ready for viewing or editing.

Example 59–2 Generated Portlet Entry for a Task Flow

```
<portlet id="adf_taskflow_WEB_INF_department_xml">
    <description>PortletBridgeApplication department</description>
    <portlet-name>PortletBridgeApplication_department</portlet-name>
    <display-name>PortletBridgeApplication department</display-name>
    <portlet-class>oracle.portlet.bridge.adf.application.ADFBridgePortlet</portlet-class>
    <init-param>
      <name>javax.portlet.faces.defaultViewId.view</name>
      <value>
        /adf.task-flow?adf.tfDoc=/WEB-INF/adfp-portlet-bridge-container.xml
        &amp;amp;adf.tfId=adfp-portlet-bridge-container&amp;amp;_fragmentTaskFlowDoc=/WEB-INF/
        department.xml&amp;amp;_fragmentTaskFlowId=department
      </value>
    </init-param>
    <supports>
      <mime-type>text/html</mime-type>
      <portlet-mode>VIEW</portlet-mode>
    </supports>
    <supported-locale>en</supported-locale>
    <portlet-info>
      <title>PortletBridgeApplication department</title>
      <short-title>PortletBridgeApplication department</short-title>
    </portlet-info>
    <supported-publishing-event id="DepartmentSelectedEvent">
      <qname xmlns:x="http://xmlns.oracle.com/adfm/contextualEvent">
        x:DepartmentSelectedEvent
      </qname>
    </supported-publishing-event>
    <supported-public-render-parameter>
      _adf_event_DepartmentSelectedEvent
    </supported-public-render-parameter>
    <container-runtime-option>
      <name>com.oracle.portlet.requireIFrame</name>
      <value>true</value>
    </container-runtime-option>
    <container-runtime-option>
      <name>com.oracle.portlet.minimumWsrpVersion</name>
      <value>2</value>
    </container-runtime-option>
</portlet>
```

Note: The `portlet.xml` file can include multiple portlets and can have a combination of pages and task flows exposed as portlets.

59.2.2.2 Creating a Portlet From a Task Flow Using the Manage Portlet Entries of Task Flows Dialog

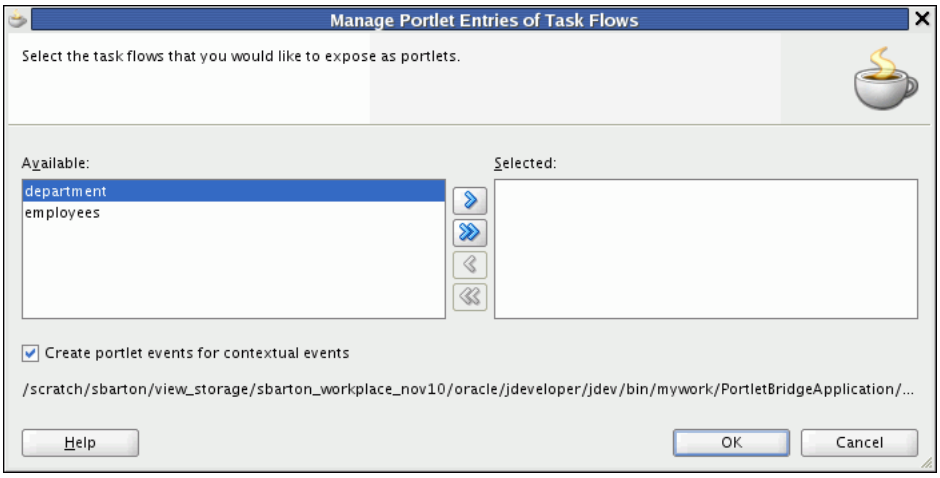
If your project includes a lot of task flows, you may find it easier to select the task flows to create as portlets from a list. You can do this using the Manage Portlet Entries of Task Flows dialog. This dialog also lets you create portlets from multiple task flows at the same time, rather than having to create them individually.

To create a portlet from a task flow using the Manage Portlet Entries of Task Flows dialog:

1. From the main menu, choose **File** and then **New.**
2. In the New Gallery, expand **Web Tier,** select **Portlets** and then **Manage Portlet Entries of Task Flows,** and click **OK.**

3. In the Manage Portlet Entries of Task Flows dialog, use the shuttle buttons to select which task flows you want to create as portlets.

Figure 59–3 The Manage Portlet Entries of Task Flows Dialog



4. Select **Create portlet events for contextual events** to create portlet events in the `portlet.xml` file for any contextual events exposed by the task flows. This option is selected by default.

   Portlet events enable a portlet to communicate with the page on which it resides and with other portlets on that page.

5. Click **OK** to create portlets for the selected task flows.

59.2.3 How to Test a JSF Portlet Using the Integrated WebLogic Server

When you have created your JSF portlet you can test it using the Integrated WebLogic Server that comes packaged with JDeveloper.

To test a JSF portlet:

1. From the main menu, choose **Run** and then **Start Server Instance.**

   It may take a few moments for the Integrated WLS to start. When the instance has started, you should see a message similar to the following in your Log panel:

   ```
   IntegratedWebLogicServer started.
   ```

   For more information, see Section 3.4, "Working with the Integrated WebLogic Server."

2. You are now ready to run or deploy your Portlet Producer application to the Integrated WLS. Because your web application and Portlet Producer application are one and the same (the Portlet Producer application is your existing web application with additional portlet artifacts), you can run or deploy your web application as you normally would or you can run or deploy your portlet following the instructions in Section 62.2.1, "How to Test JSR 286 Portlets on Integrated WebLogic Server."

   Note the distinction between Run and Deploy in the note in that section. Deploy provides a more persistent testing scenario.

3. When deployed, you can view the Producer Test Page by going to:

   ```
   http://host:port/context-root/info
   ```

Post deployment you must verify that the application works correctly as a web application before it is consumed as a Portlet Producer application. For example, verify that the page you portletized earlier works:

```
http://localhost:7101/myApp-ViewController-context-root/faces/myPage.jspx
```

4. Once you have successfully deployed the application containing the portlet, you can register it as a portlet producer with any other application. For more information see Section 64.2.1, "How to Register a WSRP Portlet Producer."

> **Note:** Because the Oracle JSF Portlet Bridge uses WSRP 2.0 features, you should register the producer using the **WSRP v2 WSDL** URL listed in the WSRP Producer Test Page.

You can continue to access the application as a regular web application or consume it as a portlet producer.

5. Now that your portlet producer is deployed and registered, you can consume your JSF portlet as you would any other portlet. For more information, see Section 64.3, "Adding Portlets to a Page."

59.2.4 What Happens at Runtime

After having created and tested the JSF portlet you can deploy the application to your production environment.

> **Note:** Ensure that you deploy your application to a Java EE container with the Oracle Portlet Container installed. The Integrated WLS has the container installed, which is why it is recommended for testing.

After successful deployment, you can access both the application and the portlet producer test page. For example, the application URL would be similar to:

```
http://host:port/appcontextroot/faces/pagename.jspx
```

And the portlet producer test URL would be similar to:

```
http://host:port/appcontextroot/info
```

For more information, see Section 64.2.1, "How to Register a WSRP Portlet Producer."

59.3 What You May Need to Know When Creating a JSF Portlet

You must code your JSF pages such that they produce markup that conforms with JSR 286 portlet markup fragment rules. If you have chosen to use Oracle ADF, the markup in your page comes mainly from the Oracle ADF Faces components, most of which naturally render markup in a style that is compatible with JSF portlets.

For those components that might cause problems in a portlet environment, the application developer must take special care. Some components generate markup that conflicts with the portlet environment and hence restricts their use. Other components may allow program control (inputs) that enable developers to introduce values that conflict with the portlet environment. In this latter case, you as the developer must be

aware of the potential to publish the page as a portlet and therefore properly encode a value.

This section includes the following subsections:

- Section 59.3.1, "General Guidelines"
- Section 59.3.2, "Portlet Guidelines"
- Section 59.3.3, "Security Guidelines"
- Section 59.3.4, "JSF Guidelines"
- Section 59.3.5, "Oracle ADF Guidelines"

59.3.1 General Guidelines

The guidelines that follow lay out the issues of which you should be aware as you code Oracle ADF pages that you may later choose to publish as portlets.

- Prior to creating a portlet from your JSF application, the application, including all of its pages and task flows, must run properly after you deploy it to Integrated WLS. If it cannot run as a regular web application, it cannot run as a portlet producer either.
- When writing an application that is supposed to run in both servlet and portlet environments, avoid casting to `HttpServlet` objects (or you get a `ClassCastException` when running as a portlet). Instead, use the abstraction provided by the Faces `ExternalContext` object. For example, instead of:

  ```
  HttpServletRequest request =
  (HttpServletRequest)FacesContext().getCurrentInstance().getExternalContext()
  .getRequest();
  ```

  Use the following:

  ```
  String localContextPath=
  FacesContext().getCurrentInstance().getExternalContext().getRequestContextPath(
  );
  ```

  To the extent that `ExternalContext` does not provide that abstraction for you, you can write servlet or portlet specific code. You can use the following method to check if the application runs as portlet:

  ```
  public static boolean isPortletRequest() {
      Map<String, Object> m =
          FacesContext.getCurrentInstance().getExternalContext().getRequestMap();
      Object phase = m.get("javax.portlet.faces.phase");
      if (phase != null) {
          return true;
      }
      else {
          return false;
      }
  }
  ```

- When consuming a JSF Portlet, the consumer application automatically renders the portlet content within an inline frame (IFRAME). The implication is that any inline popup is then confined within the IFRAME. You should take this into consideration when specifying the size of the portlet.

- If your application is secured, ensure that you have secured identity propagation as described in Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."
- Deep links are not directly supported. A by-product of the JSR 286 portlet container implementation on WSRP is that session cookie management is proxied by the consuming application rather than the client. Portlets that deep link to their full service application usually rely on shared session state to allow the transition from the current portlet context. As most applications rely on maintaining session context with a cookie, the current architecture prevents such state sharing. A deep link from the client directly to the producer server to invoke the application does not establish a session cookie between the consumer and the producer, hence a second session is established. Applications that are required to share such state must implement their own schemes for transferring the data between the two contexts. A common implementation is to write this state to a reachable location and pass a reference to this state in the deep link.
- Java EE login is not supported. Java EE applications can be configured with different authentication techniques, such as Basic and Digest. As portlets are fragments incorporated into a consumer's page, it is generally expected that direct authentication occurs between the client and the consumer, not the client and the portlet producer. As a result, these authentication techniques are not supported in JSR 286. In the JSR 286 case, Java EE authentication occurs through WS-Security mechanisms that allow the Web service consumer to be authenticated and verified by the producer, and propagate the user identity for user authentication and authorization. Published Oracle ADF artifacts must not contain login links that trigger Java EE authentication.
- For applications that take a long time to render, consider increasing the time out period when you register a producer that was created by the Oracle JSF Portlet Bridge. Note however that the time out period specified during producer registration is limited by the maximum time out period (`maximumTimeout` element) specified in `adf-config-xml`.

59.3.2 Portlet Guidelines

The portlet guidelines are as follows:

- For resources and links, you must specify the location relative to the `web-app-context-root`. Otherwise, your images and other resources cannot be found by the portlet. Do not use relative (../) path notation. Portlets in Oracle WebCenter Portal: Framework run remotely and are accessed using a SOAP protocol (WSRP). The latter means that the regular web application concept of request path does not apply in a JSR 286 container. The JSR 286 specification reflects this by mandating that all resource URLs either be absolute or context path relative.
- Do not redirect or forward a request within your JSP. JSR 286 only supports `requestDispatcher.include()`. The use of `httpServletResponse.sendRedirect()` or `requestDispatcher.forward()` results in exceptions and errors. To work properly in a portlet environment, you must implement JSF navigation rules in `faces-config.xml` or Oracle ADF task flow control flow rules.
- To minimize overall memory consumption in the application when running as a portlet, you should only store the minimal amount of data in the request scope.
- If you portletize an application that contains downloadable resources, the portlet container may rewrite the file name in such a way that it may be too long for the

browser to open or save. In such a case, you can save the resource to a different name and open the file using the appropriate program directly.

- If the portlet uses JSF view components other than ADF Faces (for example, Trinidad components), then you may want to manually change the portlet's `minimumWsrpVersion` container runtime option to `1` so that it can be registered using the WSRP V1 WSDL. For more information, see Section 61.2.4, "How to Customize the Runtime Environment."

59.3.3 Security Guidelines

The security guidelines are as follows:

- When you use the Create Portlet Entry or Manage Portlet Entries of Task Flows dialog to portletize a task flow in an Oracle ADF secured application (the security scheme being Authentication and Authorization), you must manually grant permission to the following wrapper task flow in the `jazn-data.xml` file of the producer application:

  ```
  /WEB-INF/adfp-portlet-bridge-container.xml#adfp-portlet-bridge-container
  ```

  If you do not grant the permissions, the portlet does not render.

  For example, if you have an application with two roles: authenticated-role, with view permission; and testrole, with customize, edit, grant, personalize, and view permissions, you must add the following entries inside the `<permissions>` tag of each role:

  - For authenticated-role:

    ```
    <permission>
      <class>oracle.adf.controller.security.TaskFlowPermission</class>
      <name>
      /WEB-INF/adfp-portlet-bridge-container.xml#adfp-portlet-bridge-container
      </name>
      <actions>view</actions>
    </permission>
    ```

  - For testrole:

    ```
    <permission>
      <class>oracle.adf.controller.security.TaskFlowPermission</class>
      <name>

    /WEB-INF/adfp-portlet-bridge-container.xml#adfp-portlet-bridge-container
      </name>
      <actions>customize,edit,grant,personalize,view</actions>
    </permission>
    ```

- If you are portletizing a task flow or page that has role based authorization (that is, the task flow or page has been granted certain roles), WS-Security is required to propagate the user correctly. If you set up WS-Security, the JSF portlet does not render the content due to lack of permissions.

  For information about setting up WS-Security on the producer, see Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

  When registering the producer with the consuming application, you must ensure that you set the appropriate security properties. For more information, see Section 64.2.1, "How to Register a WSRP Portlet Producer."

59.3.4 JSF Guidelines

The JSF guidelines are as follows:

- When using the `h:commandLink` JSF standard HTML component ensure that you set the following `context-param` in `web.xml` so that the JSF Reference Implementation does not generate any external JavaScript resource. This is to work around an issue in the JSF Reference Implementation where the reference to the JavaScript resource is not properly encoded.

```
<context-param>
    <param-name>com.sun.faces.externalizeJavaScript</param-name>
    <param-value>false</param-value>
</context-param>
```

59.3.5 Oracle ADF Guidelines

The Oracle ADF guidelines are as follows:

- If you are portletizing an Oracle ADF task flow, ensure that you are able to consume that task flow by dropping it onto a page as a region and running the page. This ensures that the task flow runs correctly before portletizing it.
- ADF Faces Dialog Framework is not supported. If your application includes any buttons or icons that launch secondary browser windows, then the contents of the new windows are not displayed properly when the application is run as a portlet. As such, you should avoid using these components if you plan to portletize your application. Examples of components that launch secondary windows are:
  - `<tr:inputDate>`
  - `<tr:inputColor>`
  - `<tr:popup>`
  - the `useWindow` attribute of `<af:commandButton>`
- Portletization of pages that contain Composer components is not supported.
- The `<af.fileDownloadActionListener>` component is not supported.
- Oracle ADF components/code that handle `prepareModel` must be idempotent. Any code running during the `prepareModel` phase must be rerunnable without effect to the underlying data/business logic. When running in the portlet environment, `prepareModel` is called twice during the complete JSF life cycle as opposed to being called once when running within a regular web application.

  The reason for this difference is that the Oracle JSF Portlet Bridge runs JSF in two requests not one. It is implemented as if every JSF request redirected before the render phase. The consequence of this approach is that the JSF `restoreView` phase is called both when the request is first submitted and then again when request to render is received.
- Do not access or reference request parameters from model code except in page parameters. Besides being a cleaner MVC2 implementation, following this guideline avoids problems when such artifacts are published as portlets. Where regular JSF artifacts run their entire lifecyle in a single request, the Oracle JSF Portlet Bridge runs these artifacts in two requests, as if every JSF request redirected before the render phase.

  This two phase model enables the clearing of submitted parameters before rendering in a manner that allows such clearing to be communicated all the way back to the client, which makes this request something you could bookmark. As a

result, request parameters do not exist during the render phase. As described previously, `prepareModel` is invoked again in this rendering phase. Therefore, any references to request parameters in this phase handler fail. You should avoid code like either of the following fragments:

```
<invokeAction id="doExecuteWithParams"
              Binds="ExecuteWithParams"
              Refresh="prepareModel"
              RefreshCondition="${param.id != null}"
/>

<invokeAction id="doExecuteWithParams"
              Binds="ExecuteWithParams"
              Refresh="renderModel"
              RefreshCondition="${param.id != null}"
/>
```

- Do not reference page parameters in the `prepareModel` phase. This issue relates to the same problem just described for request parameters. Generally, page parameters depend on request parameters and are evaluated during the `restoreView` phase. As this phase is called a second time during portlet rendering and request parameters are not available, such use fail. Instead, move any dependency on a page parameter value into the model before JSF transitions from its execute phases to its render phase.
- If you are portletizing an application that contains only Trinidad components (`<tr: >` tags only), then you must manually include the following libraries in your project:
  - `jdeveloper\modules\oracle.adf.view_11.1.1\adf-richclient-api-11.jar`
  - `jdeveloper\modules\oracle.adf.view_11.1.1\adf-richclient-impl-11.jar`

  This is so that the URLs to icons in the style sheet generated for the producer are encoded correctly.
- Because a portlet is a naming container, special consideration should be taken when using ADF Faces client-side APIs to find components. An example component ID:
  - When run as a regular web application: `id="demoTemplate:popup"`
  - When run as a Portlet Producer application: `id="__ns12345678:demoTemplate:popup"`

  Things to watch out for specifically are:
  - Avoid using the `AdfPage.PAGE.findComponentByAbsoluteId()` API. Use `getSource()` and `findComponent()` methods instead. For example:

```
<trh:script text="
function showPopup(event) {
    event.cancel();
    // var popup =
    //  AdfPage.PAGE.findComponentByAbsoluteId("demoTemplate:popup");
    var source = event.getSource();
    var popup = source.findComponent("popup");
    popup.show({align:"after_end", alignId:"button"});
}
"/>
```

  - Use a relative path for `clientId` instead of an absolute path (starting with a ':'). For example, use:

    ```
    <af:showPopupBehavior popupId="demoTemplate:iteratorpop"
                triggerType="mouseHover"/>
    ```

    Instead of:

    ```
    <af:showPopupBehavior popupId=":demoTemplate:iteratorpop"
                triggerType="mouseHover"/>
    ```

  - For more information, see "What You May Need to Know About Using Naming Containers" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

- If you are portletizing an Oracle ADF task flow that triggers a contextual event, the Oracle JSF Portlet Bridge creates a JSR 286 event to wrap the contextual event. Part of the JSR 286 event's payload is the wrapped contextual event payload, which is serialized. When this wrapped contextual event is delivered, the contextual event payload must be deserialized on the consumer before the contextual event can be forwarded into the consumer application. To deserialize the event, the payload class must be available on the consumer as well as on the producer. This condition is automatically met when the original payload is a Java Runtime Environment class, such as `java.lang.String`.

- If your Oracle ADF task flow includes ADFm events that do not have corresponding portlet events declared in the `portlet.xml` file, you can enable the Oracle JSF Portlet Bridge to raise these undeclared events.

  To raise undeclared events, set the following container runtime option for the portletized task flow:

  ```
  <portlet id="Application5untitled1jspx1_1">
     ...
     <container-runtime-option>
        <name>oracle.portlet.bridge.adf.raiseUndeclaredContextualEvents</name>
        <value>true</value>
     </container-runtime-option>
     ...
  </portlet>
  ```

  Setting this option to `true` means that any ADFm event raised is forwarded on as a portlet event. If this option is not provided or is set to `false`, then only events with corresponding portlet event declarations are forwarded.

  You can also enable undeclared portlet events to automatically be forwarded on from the portlet binding as ADFm events.

  In the portlet binding, set the `raiseUndeclaredContextualEvent` attribute to `true`, for example:

  ```
  <portlet id="Application5untitled1jspx1_1"
           portletInstance="/oracle/adf/portlet/WsrpPortletProducer4/ap/
                 Application5untitled1jspx_2943bd23_012e_1000_8004_0aa7c0849010"
           class="oracle.adf.model.portlet.binding.PortletBinding"
           retainPortletHeader="false"
           listenForAutoDeliveredPortletEvents="true"
           listenForAutoDeliveredParameterChanges="true"
           raiseUndeclaredContextualEvents="true"
           xmlns="http://xmlns.oracle.com/portlet/bindings"/>
  ```

Setting this option to `true` means that if a portlet event is received, a corresponding ADFm event is raised, even if there is no event declaration in the portlet binding. The name of the ADFm event is directly taken from the QName of the portlet event. This is so that ADFm events raised from the Oracle JSF Portlet Bridge have the same name as that used when the event was raised on the remote application.

- By default, URLs in the portletized page or task flow are encoded such that they are proxied by the consumer application. In some circumstances, you may not want to encode these URLs, for example, when requesting JavaScript libraries from content delivery networks or accessing resources directly from the user's browser to make use of a locally configured HTTP proxy. To bypass the URL encoding, set the `_xEncodeUrl` parameter to `false` on URLs that are passed to the Oracle JSF Portlet Bridge for encoding.

  For example, an inline frame that contains a Google Map would typically need to make use of the user's browser HTTP proxy settings to access the external `maps.google.com` server. The ADF markup to enable the component to work correctly when exposed as a portlet would look like the following:

```
<af:inlineFrame id="if1"
                shortDesc="Location of office"
                source="http://maps.google.com/maps?q=RG6+1RA&amp;
                        output=embed&amp;_xEncodeUrl=false"/>
```

  Note: Do not use this parameter to access resources that reside on the producer server. These resources must be accessed using the portlet client's proxy.

59.4 Copying a Runtime-Created Skin to a JSF Portlet Producer Application

In a WebCenter Portal application, you can create skins at runtime, using the Resource Manager. When you do this, you may encounter rendering issues on pages that include JSF portlets. This is because the skin used by the WebCenter Portal application is not available to the remote application rendering the task flow.

To rectify this issue, you must copy the runtime-created skin to the Portlet Producer application created when the task flow was portletized.

To copy a runtime-created skin to a Portlet Producer application:

1. Export the skin from the Framework application to an EAR file.

   For information about how to do this, see Section 17.4.1, "How to Download a Resource Using the Resource Manager."

2. Repackage the exported skin as a shared library (a JAR file):

   a. Extract the `transport.mar` file:

```
$ jar xvf myskin.ear
inflated: transport.mar
```

   b. Extract the metadata files:

```
$ jar xvf transport.mar
inflated: oracle/webcenter/siteresources/.../Skin.css
inflated: oracle/webcenter/siteresources/.../generic-site-resources.xml
...
```

c. Locate and view the file `generic-site-resources.xml` among the metadata files. Normally, this file is in a directory like:

```
oracle/webcenter/siteresources/scopedMD/scopeGUID/generic-site-resources.xml
```

The file should have a section that describes the exported skin, similar to the following example:

```
<resourceType name="skin" ...>
  <resource displayName="MySkin"
     metadataFile="/oracle/webcenter/siteresources/.../Skin.css" ...>
    <customAttributes>
      <customAttribute name="skinId"
         value="gsr616d879d_99e0_4bd9_8c10_98e7ea272a6a.desktop" .../>
      <customAttribute name="skinFamily"
         value="gsr616d879d_99e0_4bd9_8c10_98e7ea272a6a" ...>
      <customAttribute name="skinExtends"
         Value="webcenter-fusion-internal.desktop" .../>
    </customAttributes>
  </resource>
</resourceType>
```

d. Note the following information from `generic-site-resources.xml`:

- *`skinId`* (for example, `gsr616d879d_99e0_4bd9_8c10_98e7ea272a6a.desktop`)
- *`skinFamily`* (for example, `gsr616d879d_99e0_4bd9_8c10_98e7ea272a6a`)
- *`skinExtends`* (for example, `webcenter-fusion-internal.desktop`)

e. Build the directory structure for the JAR file:

```
$ mkdir META-INF
```

f. Copy the `Skin.css` file into the `META-INF` directory:

```
$ cp oracle/webcenter/siteresources/.../Skin.css META-INF
```

g. Create a new file called `trinidad-skins.xml` under the `META-INF` directory:

```
$ edit META-INF/trinidad-skins.xml
```

h. Add the following XML to the new `trinidad-skins.xml` file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<skins xmlns="http://myfaces.apache.org/trinidad/skin">
  <skin>
    <style-sheet-name>Skin.css</style-sheet-name>
    <id>skinId</id>
    <family>skinFamily</family>
    <extends>skinExtends</extends>
    <render-kit-id>org.apache.myfaces.trinidad.desktop</render-kit-id>
  </skin>
</skins>
```

where ***`skinId`***, ***`skinFamily`***, and ***`skinExtends`*** are the values noted earlier.

i. Package the JAR file:

```
$ jar cvf myskin.jar META-INF
adding: META-INF/trinidad-skins.xml(in = 359) (out= 171)(deflated 52%)
adding: META-INF/Skin.css(in = 5560) (out= 1413)(deflated 74%)
```

The JAR file should contain two files: `Skin.css` and `trinidad-skins.xml`.

3. Copy the new `myskin.jar` file to the Portlet Producer application.

   The easiest way to do this is to copy the file to the `WEB-INF/lib` directory of the Portlet Producer web application.

60

Creating Portlets with the Portlet Wizard

This chapter explains how to create Java portlets based on the Java Portlet Specification (JSR 286) or the Oracle Portal Developer Kit-Java (PDK-Java) in a WebCenter Portal Portlet Producer application, using the Create JSR 286 Java Portlet and Create Oracle PDK-Java Portlet wizards.

This chapter includes the following sections:

- Section 60.1, "Introduction to Java Portlets"
- Section 60.2, "Creating Java Portlets"
- Section 60.3, "Troubleshooting Java Portlets"

60.1 Introduction to Java Portlets

This section includes the following sections:

- Section 60.1.1, "Introduction to Standards-Based Java Portlets"
- Section 60.1.2, "Introduction to PDK-Java Portlets"

60.1.1 Introduction to Standards-Based Java Portlets

Historically organizations engaged in portal development projects found application integration to be a major issue. Users developed portlets using proprietary APIs for a single portal platform and often faced a shortage of available portlets from a particular portal vendor. All this changed with the introduction of the following standards:

- Web Services for Remote Portlets (WSRP)
- Java Specification Request (JSR) 286

These two standards enable the development of portlets that interoperate with different portal products, and therefore widen the availability of portlets within an organization. This wider availability, in turn, dramatically increases an organization's productivity when building portals.

WSRP is a web services standard that enables the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container and any WSRP portal. WSRP defines the following:

- Web Services Definition Language (WSDL) interface for the invocation of WSRP services
- Markup fragment rules for markup emitted by WSRP services
- The methods to publish, find, and bind WSRP services and metadata

JSR 286 is a specification that defines a set of APIs to enable interoperability between portlets and portals, addressing the areas of aggregation, personalization, presentation, and security. JSR 286 defines container services which provide the following:

- A portlet API for coding portlet functionality
- The URL-rewriting mechanism for creating user interaction within a portlet container
- The security and personalization of portlets
- Inter-portlet communication using portlet events and public render parameters

Oracle actively participates in portlet standards including those produced by the OASIS (WSRP) and JCP (JSR 168, 286, 301, 329, and so on) standards organizations.

> **Note:** HTML forms can be submitted using either the `get` or `post` method, but the WSRP standard only requires the consumer to use the `post` method. Support of the `get` method is optional according to the standard. Since application consumers are not required to support the `get` method, Oracle recommends that you use the `post` method when developing your portlets.

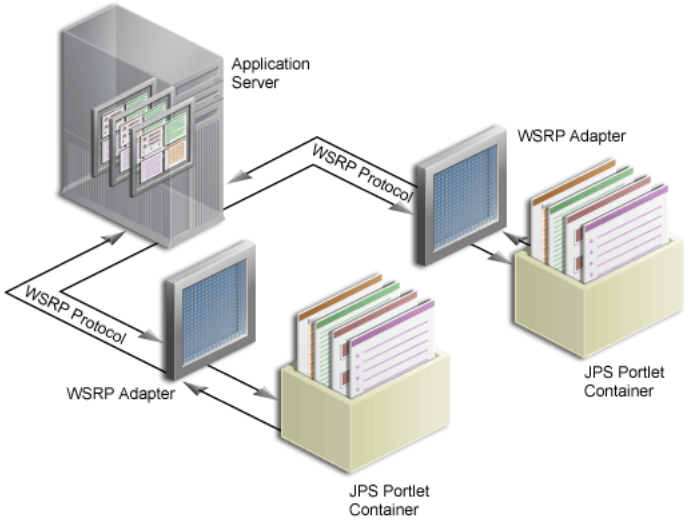
The Relationship Between WSRP and JSR 286

WSRP is a communication protocol between a portlet client, for example, a WebCenter Portal: Framework application, and a portlet container running on a remote server. JSR 286 describes the Java Portlet API for running Portlet Producer applications. Combining these standards enables developers to integrate their applications from any internal or external source as portlets with WSRP portals. Building pages becomes as simple as selecting portlets from the JDeveloper Resource Palette or Application Resources pane of the Application Navigator.

Figure 60–1 shows the architecture of the WSRP specification.

> **Note:** Figure 60–1 illustrates the use of JSR 286 portlets with WSRP.

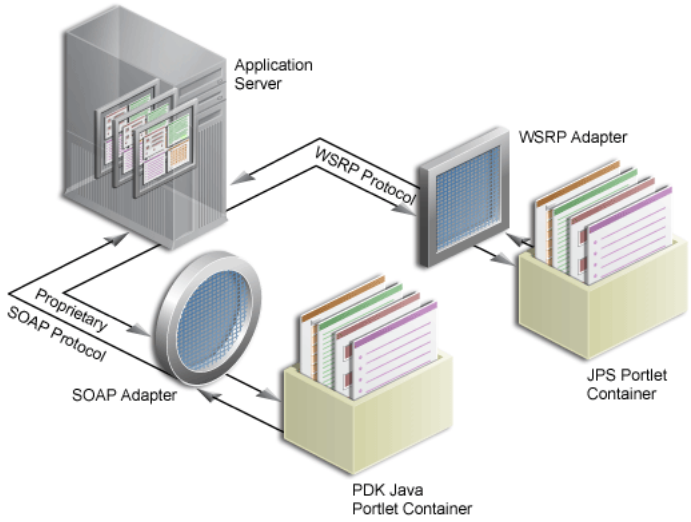
Figure 60–1 WSRP Specification Architecture



WebCenter Portal: Framework can support communication between the Framework application and both the Java Portlet APIs and our existing APIs (PDK-Java).

Figure 60–2 shows the architecture of the WSRP support. Notice that the JSR 286-compliant portlet container uses the WSRP protocol for communication and the PDK-Java portlet container uses Oracle's proprietary SOAP protocol for communication.

Figure 60–2 WebCenter Portal Portlet Architecture



For a description of how JSR 286 security concepts are exposed through WSRP, see Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

60.1.2 Introduction to PDK-Java Portlets

PDK-Java gives you a framework to simplify the development of Java portlets by providing commonly required utilities and enabling you to leverage existing development skills and application components such as JSPs, servlets, and static HTML pages. PDK-Java also enables you to create portlets without having to deal directly with the complexity of communications between WebCenter Portal: Framework and producers.

The PDK-Java framework is divided into the following areas:

- The **Producer Adapter** insulates the developer from the HTTP syntax defined by WebCenter Portal: Framework for communication with web producers. It translates the information passed between WebCenter Portal: Framework and your Java web producer. Without an adapter, your producer would not only manage portlets, but it would also have to communicate this information directly to WebCenter Portal: Framework in the expected language. The adapter eliminates the need for your web producer to understand the portal language and vice-versa.
- The **Producer Interface** defines the APIs (functions) required by your Java implementation to integrate with the Producer Adapter. The Producer Adapter receives messages from the Framework application, translates them into calls to the Producer Interface, and translates the producer's response into a format that the application can understand. The Producer Interface contains a set of Java classes that define the methods your producer implements and, in often, provides a standard implementation. Some of the primary classes are as follows:

  - `ProviderDefinition` (`oracle.portal.provider.v2.ProviderDefinition`)
  - `ProviderInstance` (`oracle.portal.provider.v2.ProviderInstance`)
  - `PortletDefinition` (`oracle.portal.provider.v2.PortletDefinition`)
  - `PortletInstance` (`oracle.portal.provider.v2.PortletInstance`)
  - `ParameterDefinition` (`oracle.portal.provider.v2.ParameterDefinition`)
  - `EventDefinition` (`oracle.portal.provider.v2.EventDefinition`)
- The **Producer Runtime** provides a base implementation that follows the specification of the Producer Interface. The Producer Runtime includes a set of default classes that implement each of the Producer Interfaces and enables you to leverage the rendering, personalization, and security frameworks provided with PDK-Java. These classes and the associated frameworks simplify the development of a producer by implementing common functions for WebCenter Portal: Framework requests and providing a declarative mechanism for configuring the producer. Using the Producer Runtime, you can focus your development efforts on the portlets themselves rather than the infrastructure needed to communicate with the Framework application. If the standard behavior of the Producer Runtime does not meet your requirements, then you can easily extend or override specific behaviors. Some of the primary classes are as follows:
  - `DefaultProviderDefinition` (`oracle.portal.provider.v2.DefaultProviderDefinition`)
  - `DefaultProviderInstance` (`oracle.portal.provider.v2.DefaultProviderInstance`)
  - `DefaultPortletDefinition` (`oracle.portal.provider.v2.DefaultPortletDefinition`)
  - `DefaultPortletInstance` (`oracle.portal.provider.v2.DefaultPortletInstance`)
  - `PortletRenderer` (`oracle.portal.provider.v2.render.PortletRenderer`)
  - `PortletPersonalizationManager` (`oracle.portal.provider.v2.personalize.PortletPersonalizationManager`)
  - `PortletSecurityManager` (`oracle.portal.provider.v1.http.DefaultSecurityManager`)
- The **Producer Utilities** provide methods for simplifying the rendering of portlets. The utilities include methods for constructing valid links (`hrefs`), rendering the portlet's container (including the header), rendering HTML forms that work within a page, and supporting portlet caching.

For more information about the PDK-Java, see the *Oracle Fusion Middleware Java API Reference for Oracle PDK Java*.

60.2 Creating Java Portlets

Before you begin you should make sure:

- You are familiar with portlet terminology such as portlet modes. For more information, see Chapter 58, "Overview of Portlets."
- You are familiar with Oracle JDeveloper and know how to build and deploy Java components using it.
- The application in which you are going to create your JSR 286 and PDK-Java portlets is either a Portlet Producer application or any application scoped for portlet creation (any application except for one using WebCenter Portal's Framework application template).

> **Note:** The figures in this section were taken with the default Look and Feel (Oracle) and Theme (Fusion Blue) settings in JDeveloper. If you have changed these settings, then what you see on your screen may vary slightly, but the content and functionality remains the same. To change the Look and Feel and Theme settings, select **Preferences** from the **Tools** menu, and then select **Environment.**

This section includes the following subsections:

- Section 60.2.1, "How to Create a JSR 286 Java Portlet"
- Section 60.2.2, "What Happens When You Create a JSR 286 Java Portlet Using the JDeveloper Wizard"
- Section 60.2.4, "How to Create a PDK-Java Portlet"
- Section 60.2.5, "What Happens When You Create a PDK-Java Portlet"
- Section 60.2.6, "After You Create an Oracle PDK-Java Portlet Using the JDeveloper Wizard"

60.2.1 How to Create a JSR 286 Java Portlet

Using the Create JSR 286 Java Portlet wizard in JDeveloper you can expose your portlet over WSRP 2.0 quickly and easily. This wizard supports both WSRP 1.0 and WSRP 2.0.

In the Create JSR 286 Java Portlet wizard, you can choose which portlet modes you want to implement and the implementation method (JSP, HTTP servlet, Java class, or HTML) to use for each mode. The wizard then creates a simple implementation for each of the selected modes.

To create a JSR 286 Java portlet using the JDeveloper wizard:

1. In the JDeveloper Application Navigator, open the application under which you want to create your portlet.

   The application must be scoped for portlet creation. An easy way to achieve this is to use the Portlet Producer Application template.

   > **Note:** Application built using WebCenter Portal's Framework application template are *not* scoped for portlet creation.

2. Right-click the project under which you want to create your portlet (for example **Portlets**), and choose **New.**

3. In the New Gallery, expand **Web Tier,** select **Portlets** and then **Standards-based Java Portlet (JSR 286),** and click **OK.**

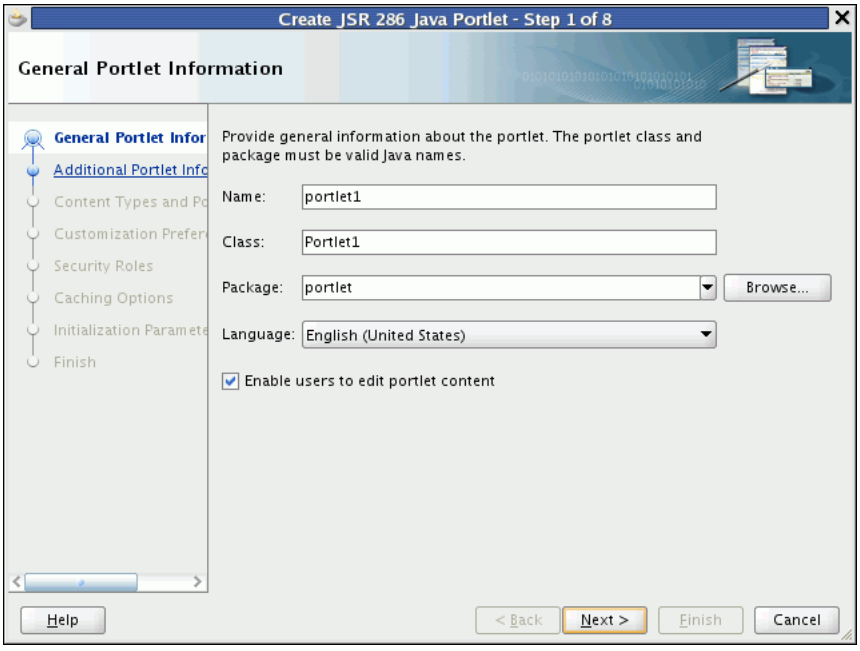
Note: If the project already contains JSR 286 portlets, you can also create a new portlet by:

- Right-clicking `portlet.xml` and choosing **Add Portlet**.
- Opening `portlet.xml`, clicking the **Design** tab, and clicking the **Add** icon on the **Portlets** tab.

---

4. On the General Portlet Information page of the Create JSR 286 Java Portlet wizard (Figure 60–3), replace the default name provided in the **Name** field with one that better describes the purpose of the portlet.

Figure 60–3 The General Portlet Information Page



5. In the **Class** field, enter a name for the class for the portlet. You can accept the default name provided or supply your own. If you supply your own name, it must be a valid Java name.

6. From the **Package** enter the name of the package in which to create the class, or select the package from the dropdown list.

   Click the **Browse** button to find packages within the project, if required. If you do not select a specific package, the wizard uses the default package of the project.

7. From the **Default Language** dropdown list, select the default language that your portlet supports. The wizard uses English by default.

8. Select **Enable users to edit portlet content** if you want your portlet to support Edit mode. In the wizard, this option is selected by default.

   Edit mode enables users to personalize the portlet at runtime. For more information, see Section 58.1.3.2, "Edit Mode."

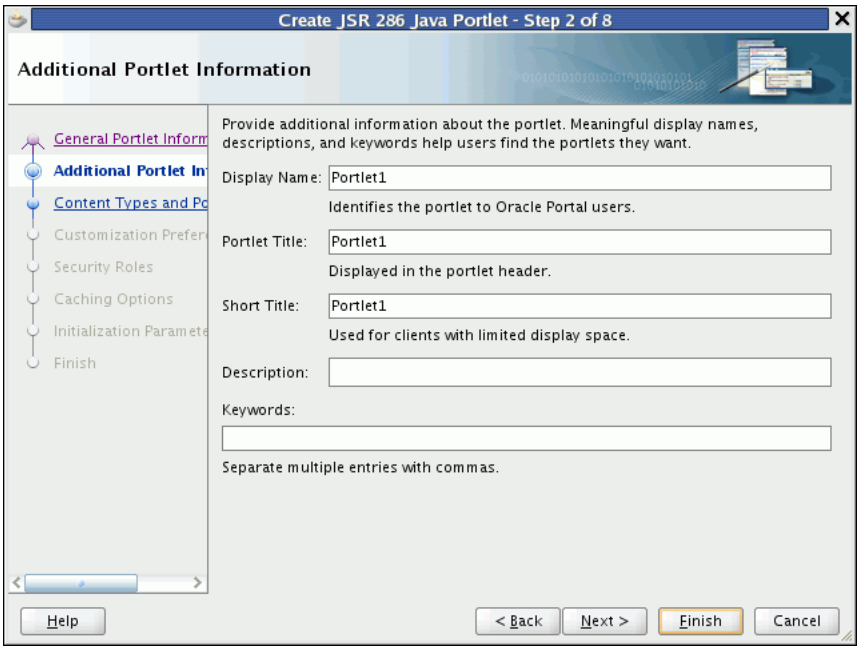
If you select this option, you can specify implementation details for the portlet's Edit mode later on in the wizard.

9. Click **Next**.

10. On the Additional Portlet Information page (Figure 60–4), in the **Portlet Title** field, enter a descriptive title for your portlet.

    The portlet title is displayed in the Resource Palette or Application Resources panel, so make the title something to help users decide whether the portlet is useful to them. The portlet title is also displayed on the portlet header when the portlet appears on a page.

Figure 60–4 The Additional Portlet Information Page



11. The **Display Name**, **Short Title**, **Description**, and **Keyword** attributes are not implemented in Framework applications. You do not need to enter any values for these fields unless your portlet is likely to be consumed by other applications.

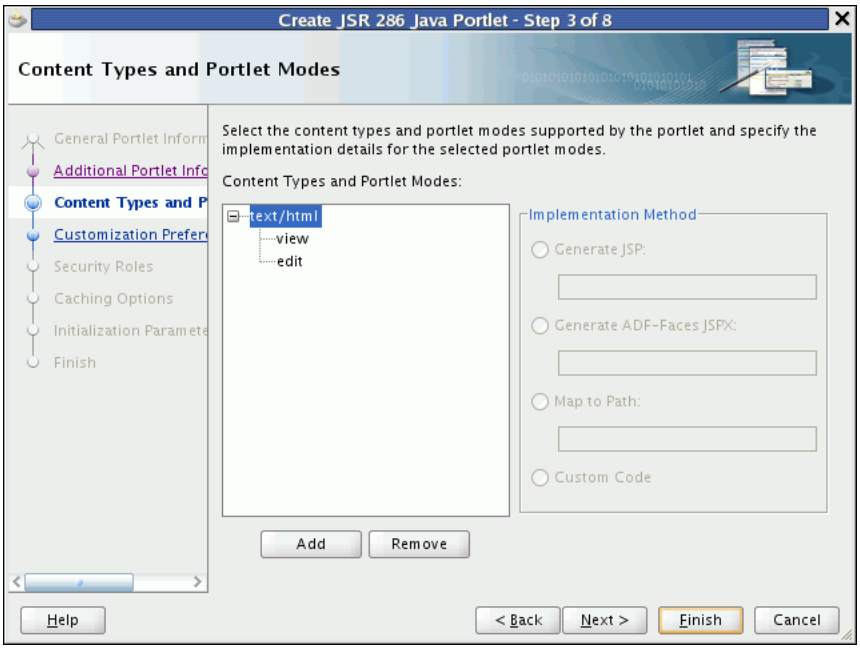
    For example, Oracle Portal uses the portlet display name in the Portlet Repository.

12. At this point in the wizard, you can click **Finish** to create the portlet immediately, using the default values for all remaining settings.

    To provide additional details for your portlet, click **Next** and follow the remaining steps.

13. On the Content Types and Portlet Modes page (Figure 60–5), in the **Content Types and Portlet Modes** list, select **view**.

Figure 60–5 The Content Types and Portlet Modes Page



14. In the Implementation Method section, select how you want to implement View mode for your portlet (for more information about View mode, see Section 58.1.3.1, "View Mode"):

    - Select **Generate JSP** to generate a skeleton JSP file for the portlet mode. Enter a name for the JSP in the corresponding field, or accept the default.

      When you complete the wizard, the generated JSP displays in the Application Navigator where you can select it for further development. This is the default selection for all portlet display modes.

    - Select **Generate ADF-Faces JSPX** to generate a page to which you can add ADF-Faces components. Enter a name for the JSPX in the corresponding field, or accept the default.

      Note: If you choose this option, the portlet implementation class is created as a subclass of `oracle.portlet.bridge.adf.application.ADFBridgePortlet`, instead of as a subclass of `javax.portlet.GenericPortlet`. That is, the wizard generates a Portlet Producer application which uses the Oracle JSF Portlet Bridge. For more information about the Oracle JSF Portlet Bridge, see Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge."

    - Select **Map to Path** to map the portlet mode to a web resource in the Portlet Producer application, such as a page. The resource is not generated by the wizard. Enter the name and path in the corresponding field. You can create this resource after completing the wizard if necessary.

      With this selection, you must write the targeted resource or file yourself. The target could be, for example, a JSP, a servlet, or an HTML file. This selection enters code in the generated portlet Java class that routes requests for the given mode to the specified target.

- Select **Custom Code** to implement the portlet mode through a custom coded object. You must create this object later. This selection generates a skeleton method to render content (`private void do`*`MODE_NAMECONTENT_TYPE`*) in the generated portlet java class. You must update this code to render useful content.

Note: If you want this portlet mode to support a different content type, for example text/xml, see Step 17.

---

15. If you selected **Enable users to edit portlet content** on the first page of the wizard, select **edit** and then select the implementation method for Edit mode (for information about Edit mode, see Section 58.1.3.2, "Edit Mode").
16. To implement another portlet mode for your portlet:
    a. In the **Content Types and Portlet Modes** list, select an existing mode (for example, **view**) under the appropriate content type (for example, **text/html**).
    b. Click **Add**.
    c. In the Portlet Modes dialog, move the required modes to the **Selected** list and click **OK**.

       The Portlet Modes dialog lists the standard modes supported by JSR 286 and the extended modes supported by WebCenter Portal.

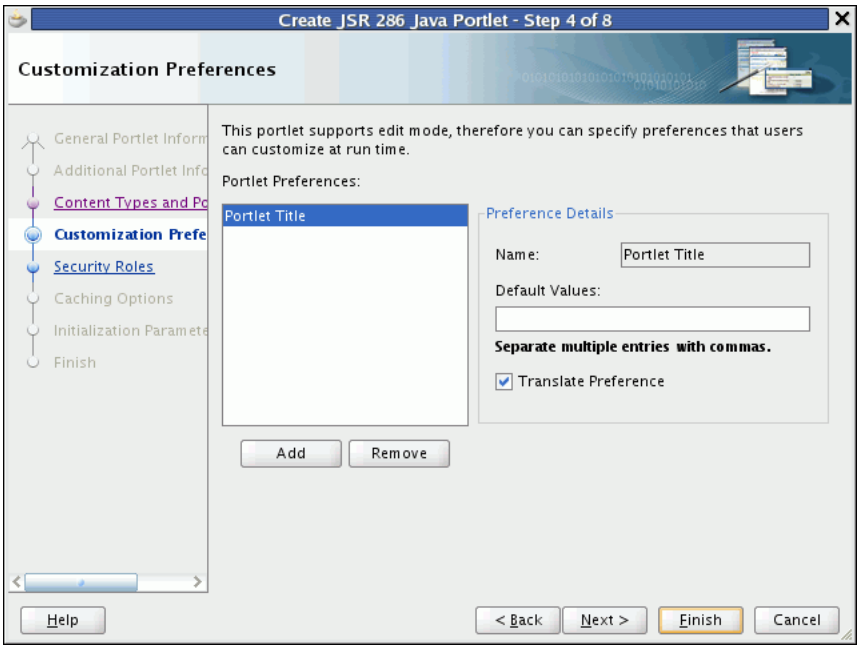
       For more information, see Section 58.1.3, "Portlet Modes."
    d. Select each of the portlet modes and specify the implementation method to use for rendering content, as described in step 14.
17. To add a new content type:
    a. In the **Content Types and Portlet Modes** list, select an existing content type (for example, **text/html**).
    b. Click **Add**.
    c. In the Content Types dialog, move the required content types to the **Selected** list and click **OK**.
18. Click **Next.**

    If you selected **Enable users to edit portlet content** on the General Portlet Information page earlier in the wizard, then you can create customization preferences to enable users of the portlet to specify values for the portlet at runtime. Go to step 19.

    If you did not select this option, go to step 24.
19. On the Customization Preferences page (Figure 60–6), click **Add** to add a new customization preference to the portlet.

    By default, the wizard includes a customization preference to enable users to customize the portlet title.

Figure 60–6 The Customization Preferences Page



20. In the Add New Preference dialog, in the **Name** field, enter a name for the new preference.

    The name must be unique in the portlet. Use only letters, numbers, and the underscore character.

21. In the **Default Values** field, enter one or more default values for the new preference. Separate multiple values with commas.

22. To make the preferences translatable, select the **Translate Preference** checkbox and then click **OK.**

    JDeveloper generates a resource bundle class for translatable preferences, with strings for which you can obtain translations. At runtime, the portlet references the resource bundle entries.

    Note: The preference name is always translated, but there is not always a requirement to translate the default values. For example, if the value is an integer, then no translation is needed.

    ---

23. Repeat the preceding steps to add more preferences. When you are done click **Next.**

24. On the Security Roles page (Figure 60–7), to add an existing security role to your portlet, select the security role and move it to the **Selected** list.

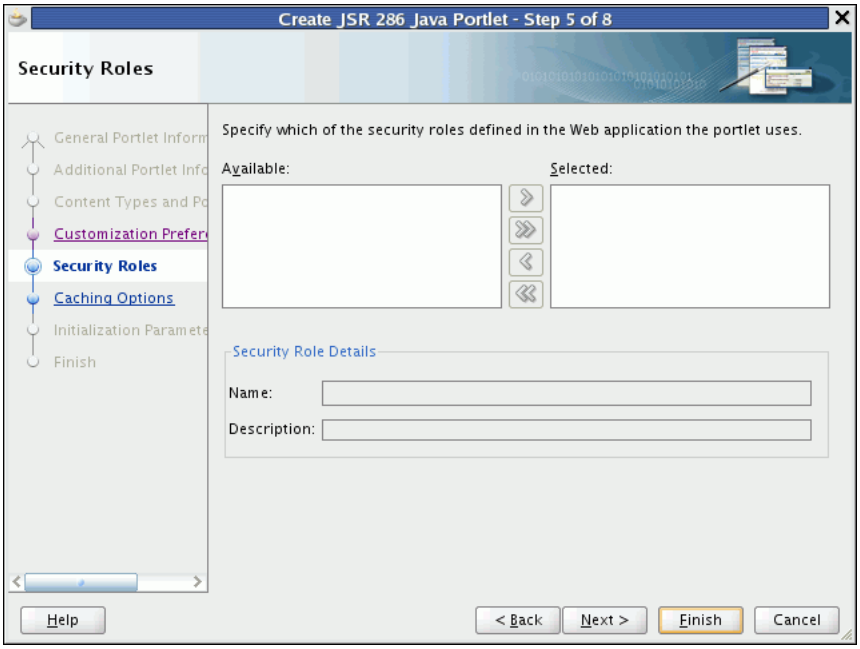
    Security roles enable you to set tiered levels of access to the portlet. For example, a *View* user can view the portlet but cannot edit it; a *Customize* user can customize portlet settings; a *Manage* user can perform all available functions associated with the portlet.

    The **Available** list displays the security roles defined for the application in which you are creating the portlet. Moving a security role to the **Selected** list creates a reference of the security role in the application's portlet deployment file

(`portlet.xml`) that refers to the security role in the application's web deployment file (`web.xml`).

You can create new security roles for the application by editing `web.xml`. For more information, see the JDeveloper online help.

Figure 60–7 The Security Roles Page



25. Click **Next.**

26. On the Caching Options page (Figure 60–8), select **Cache Portlet** to enable expiry-based caching for your portlet.

    Selecting this option indicates that portlet caching is managed by the portlet container. The portlet itself may choose to cache content for any given response. The settings on this page apply only when the portlet itself does not specify a caching condition for a response.

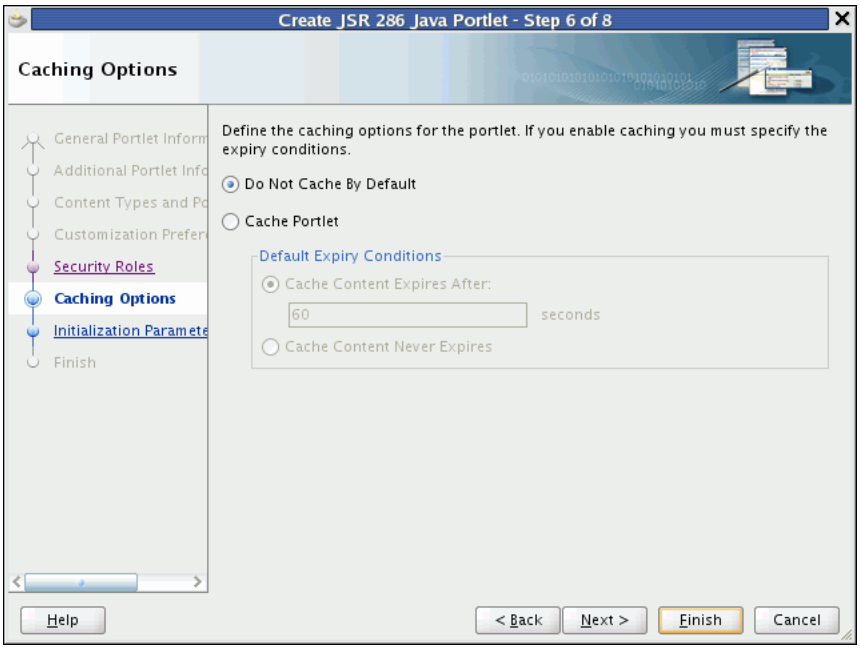
    If you do not want any default caching for this portlet, choose **Do Not Cache By Default.** In this case, the wizard actually sets a cache duration of 0 seconds. As stated earlier, this cache setting only comes into play when the portlet itself does not specify a caching condition for a response.

    If you choose no caching here and you later decide to implement default caching for the portlet, then you can change the cache duration value in the `portlet.xml` file, which is generated by the wizard, to a number greater than zero.

    For more information, see Section 61.2.9.1, "Implementing Expiry-Based Caching."

    For information about how to implement validation-based caching, see Section 61.2.9.2, "Implementing Validation-Based Caching."

Figure 60–8 The Caching Options Page



27. If you chose to cache the portlet, in the Default Expiry Conditions section, select:
    - **Cache Content Expires After [ ] seconds** to expire the cached portlet content after a certain amount of time. Specify the time limit in the corresponding field.
    - **Cache Content Never Expires** to never expire the cached portlet content. You may want to select this option if the portlet contains static content that is unlikely to change.
28. Click **Next.**
29. On the Initialization Parameters page (Figure 60–9), you can add initialization parameters to your portlet.

    Initialization parameters provide the application developer, who decides what goes into the `.war` file, an alternative to JNDI variables for configuring the behavior of all of the different components of the application (for example, servlets and portlets) in a compatible way. These initialization parameters are added to the `portlet.xml` file.

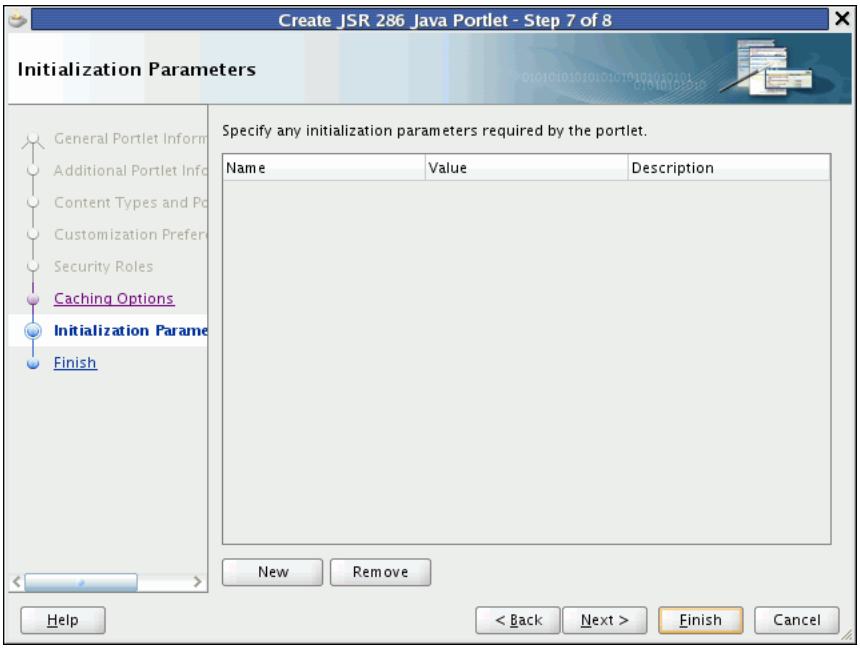
    a. Click **New** to add a new initialization parameter to the portlet.

    b. In the newly added row, double-click each field to provide a **Name**, default **Value**, and **Description** for the parameter.

    c. Repeat these steps to add more initialization parameters.

    d. When you are done, click **Finish**.

Figure 60–9 The Initialization Parameters Page



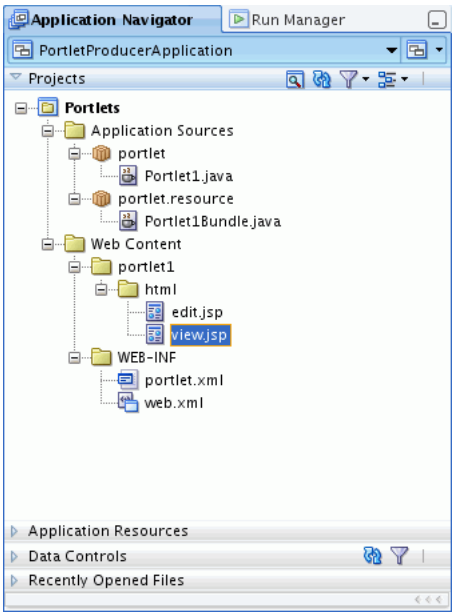
60.2.2 What Happens When You Create a JSR 286 Java Portlet Using the JDeveloper Wizard

When you use the JDeveloper wizard to create a JSR 286 Java portlet, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- Two Java classes:
  - *portletName*`.java` is invoked by the portlet container and contains all the methods required by the portlet standards.
  - *portletname*`Bundle.java` contains all the translation strings for the portlet.
- `portlet.xml` is the portlet deployment descriptor file for the application.
- `web.xml` is the web deployment descriptor file for the application.
- Files for each portlet mode you selected for the portlet:
  - If you selected **Generate JSP** for the portlet mode, a JSP page is created for the mode, for example, `view.jsp`.
  - If you selected **Generate ADF-Faces JSPX**, a JSPX page is created for the mode, for example, `view.jspx`. You can add Faces components to this page.
  - If you selected **Map to Path,** no additional files are created as the code for the portlet mode resides in an existing resource. Code is added to the portlet's Java class to route requests to the specified target.
  - If you selected **Custom Code**, no additional files are created, but the code for the portlet mode resides in the portlet's Java class.

You can see all these files in the Application Navigator, as shown in Figure 60–10.

Figure 60–10 Files Generated for a JSR 286 Java Portlet



60.2.3 After You Create a JSR 286 Java Portlet Using the JDeveloper Wizard

The next step is to extend the sample code with your own business logic to implement the desired functionality and features for your portlet. For more information, see the JSR 286 specification at:

`http://jcp.org/en/jsr/detail?id=286`

You can change the portlet's settings by editing the `portlet.xml` file. JDeveloper provides an Overview Editor to enable you to easily edit `portlet.xml`. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."

60.2.4 How to Create a PDK-Java Portlet

Using the Create Oracle PDK-Java Portlet wizard in JDeveloper you can quickly and easily create PDK-Java portlets. You can choose which portlet modes you want to implement and the implementation method (JSP, HTTP servlet, Java class, or HTML) to use for each mode. The wizard then creates a simple implementation for each of the selected modes.

To create a PDK-Java portlet using the JDeveloper wizard:

1. In the JDeveloper Application Navigator, open the application under which you want to create your portlet.

   The application must be scoped for portlet creation. An easy way to achieve this is to use the Portlet Producer application template. Applications build using WebCenter Portal's Framework application template are not scoped for portlet creation.

2. Right-click the project under which you want to create your portlet, and choose **New.**

> **Note:** To create the portlet in an existing producer, right-click the producer's `provider.xml` file and choose **Add Portlet.** This takes you directly to the General Portlet Information page of the Create Oracle PDK-Java Portlet wizard (step 8).

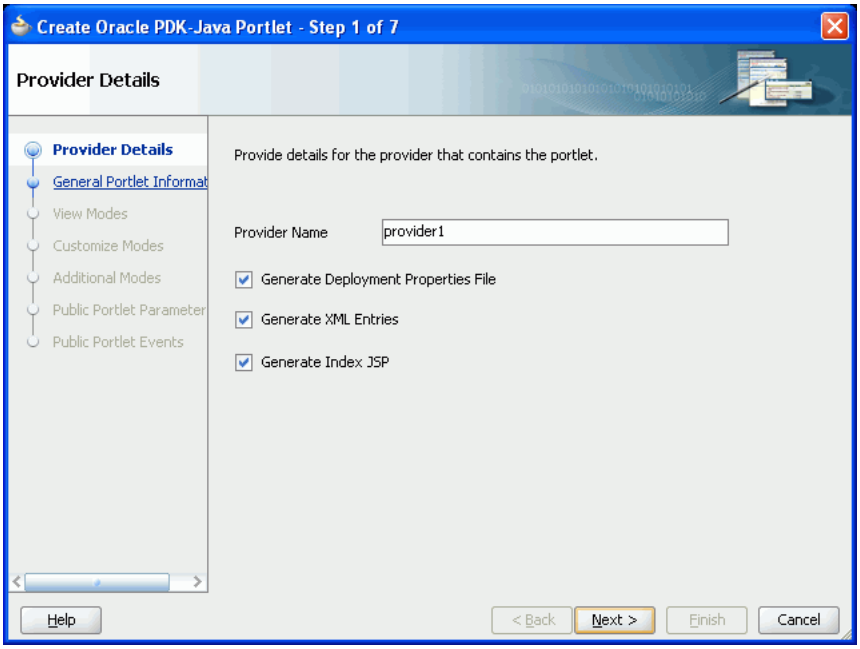
3. In the New Gallery, expand **Web Tier,** select **Portlets** and then **Oracle PDK-Java Portlet,** and click **OK.**

> **Note:** Selecting **Oracle PDK-Java Portlet** opens the wizard for creating PDK-Java portlets. Selecting **Standards-based Java Portlet** opens the wizard for creating JSR 286-compliant portlets.

4. On the Provider Details page of the Create Oracle PDK-Java Portlet wizard (Figure 60–11), enter a name for the new producer to contain your portlet. This name must be unique within the project.

   In the PDK-Java, the term provider is used instead of producer. A provider is the same thing as a producer.

Figure 60–11 The Provider Details Page



5. Select **Generate Deployment Properties File.**

   This automatically generates two `.properties` files:

   - *`serviceID`*`.properties` defines properties for a producer with that service ID. The service ID has the same value as the producer name.
   - `_default.properties` is a default properties file. A producer application may have multiple producers, each with its own service ID. On registration, if no service ID is defined, then the default properties file is used.

6. Select **Generate XML Entries.**

This automatically generates a producer definition file (`provider.xml`) for the producer that contains details of the portlets belonging to the producer, including those generated by the wizard.

7. Select **Generate Index JSP** and then click **Next.**

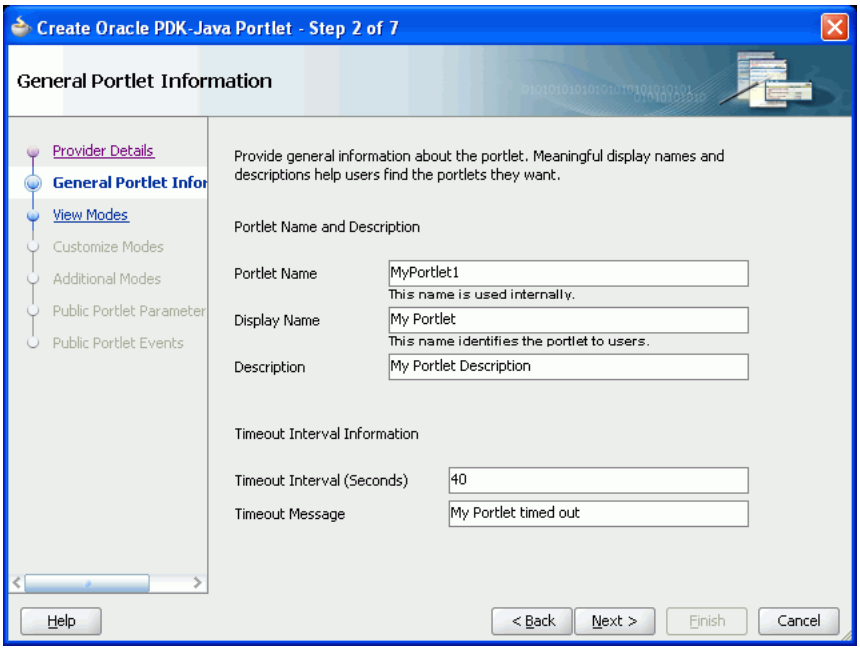
   This automatically generates an `index.jsp` file that lists all the producers that reside in the application with hyperlinks that enable easy access to producer test pages.

8. On the General Portlet Information page (Figure 60–12), enter a name and display name for your portlet.

   The name is used internally and is not exposed to users. The display name is displayed to users in portlet selection lists, such as the Component Palette.

   The description is not implemented in Framework applications so you do not need to enter a value in this field unless your portlet is likely to be consumed by other applications, such as Oracle Portal.

Figure 60–12 The General Portlet Information Page



9. In the **Timeout Interval (Seconds)** field, enter the number of seconds to allow for rendering the portlet.

10. In the **Timeout Message** field, enter a message to display if the rendering of the portlet exceeds the timeout interval specified and then click **Next.**

11. On the View Modes page (Figure 60–13), under Show Page, from the **Implementation Style** dropdown list, select the implementation style to use for the portlet's Shared Screen mode.

    - Select **JSP** to implement the portlet's Shared Screen mode as a JavaServer Page. In the **File Name** field, enter the name of the file to be generated by the wizard.
    - Select **HTTP Servlet** to implement the portlet's Shared Screen mode as an HTTP servlet. In the **Package Name** field, enter the name of the package that

contains the HTTP servlet. In the **Class Name** field, enter the Java class to be referenced with the portlet's Shared Screen mode.

- Select **HTML File** to implement the portlet's Shared Screen mode as an HTML file. In the **File Name** field, enter the name of the file to be generated by the wizard. Note that, when you choose **HTML File,** the following code is added inside the `<renderer>` element of your `provider.xml` file:

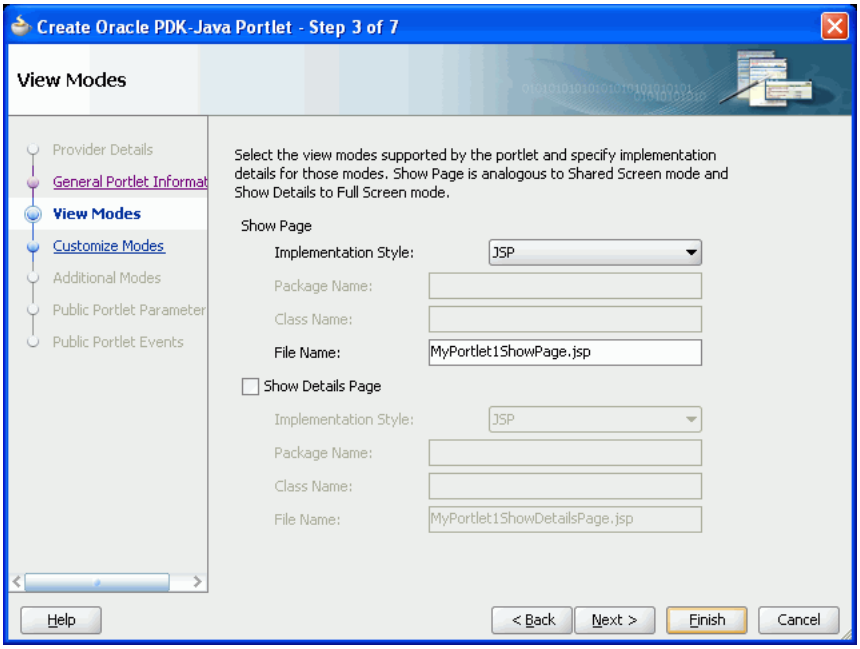
```
<showPage class="oracle.portal.provider.v2.render.http.ResourceRenderer">
    <resourcePath>provider_id/portlet_name/file.html</resourcePath>
    <contentType>content_type</contentType>
    <charSet>char_set</charSet>
 </showPage>
```

  `charSet` indicates the character set that the producer must use to encode the HTML page. The default character set is determined by JDeveloper preferences. If you require a different character set, you must update this element of `provider.xml` accordingly.

- Select **Java Class** to implement the portlet's Shared Screen mode as a Java class. In the **Package Name** field, enter the name of the package that contains the Java class. In the **Class Name** field, enter the name of the Java class.

For more information about Shared Screen mode, see Section 58.1.3.1, "View Mode."

Figure 60–13 The View Modes Page



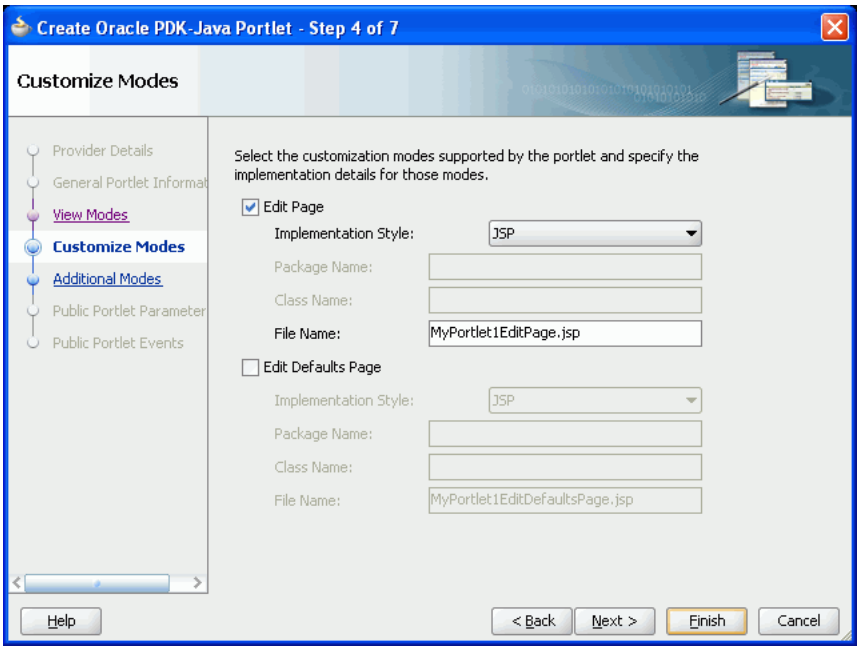
12. To implement Full Screen mode for your portlet, select **Show Details Page** and then select an implementation style as described for Full Screen mode in step 11.
13. At this point in the wizard, you can click **Finish** to create the portlet immediately, using the default values for all remaining settings.

    To provide additional details for your portlet, click **Next** and follow the remaining steps.

14. On the Customize Modes page (Figure 60–14), **Edit Page** is selected by default. To implement Edit mode for your portlet, select an implementation style as described for Shared Screen mode in step 11. If you do not want to implement Edit mode, then deselect **Edit Page.**

    For more information about Edit mode, see Section 58.1.3.2, "Edit Mode."

Figure 60–14 The Customize Modes Page



15. To implement Edit Defaults mode for your portlet, select **Edit Defaults Page**, and then select an implementation style as described for Shared Screen mode in step 11.

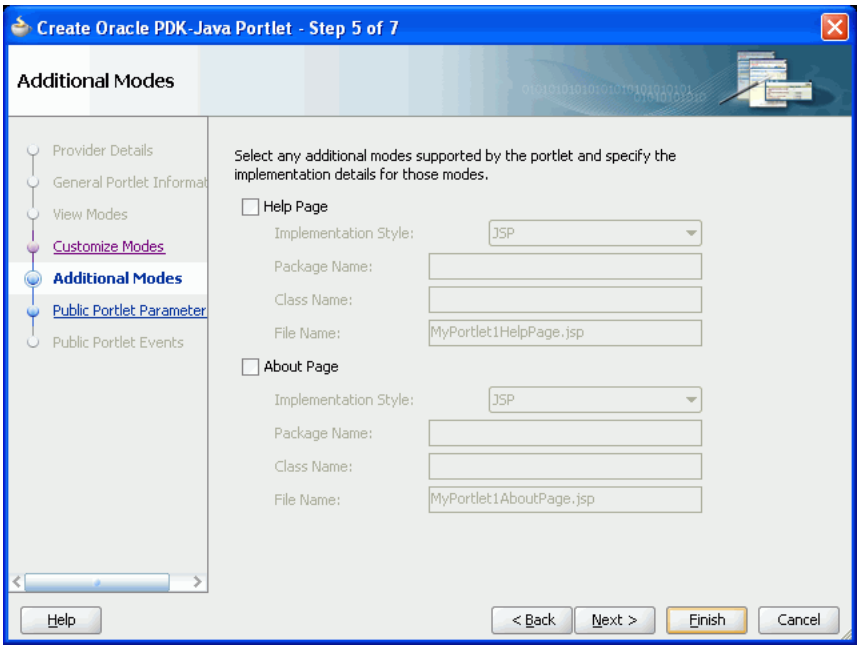
    For more information about Edit Defaults mode, see Section 58.1.3.3, "Edit Defaults Mode."

16. Click **Next.**

17. On the Additional Modes page (Figure 60–15), to implement Help mode for your portlet, select **Help Page**, and then select an implementation style as described for Shared Screen mode in step 11.

    For more information about Help mode, see Section 58.1.3.4, "Help Mode."

Figure 60–15 The Additional Modes Page



18. To implement About mode for your portlet, select **About Page**, and then select an implementation style as described for Shared Screen mode in step 11.

    For more information about About mode, see Section 58.1.3.5, "About Mode."

19. Click **Next.**

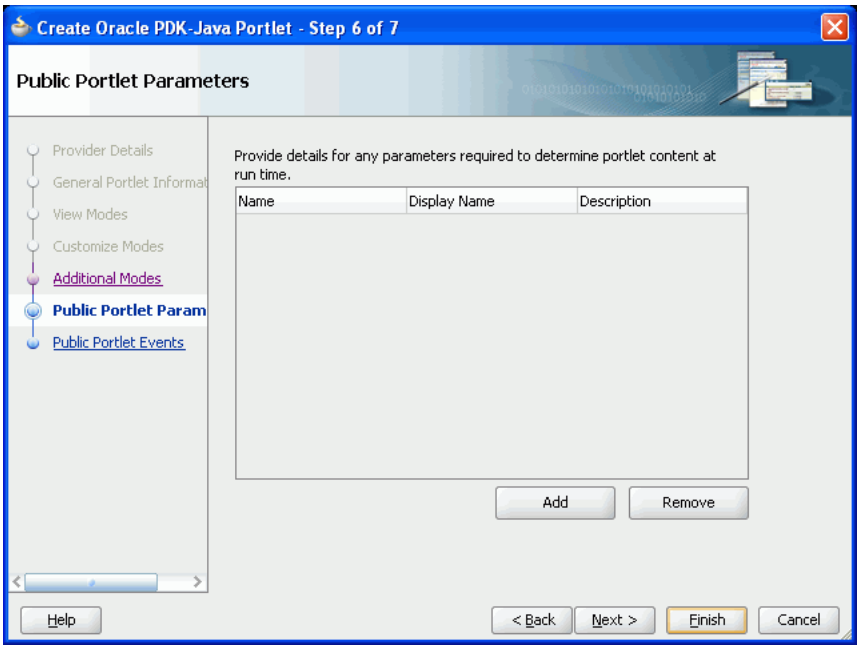
20. On the Public Portlet Parameters page (Figure 60–16), click **Add** to add a public parameter to your portlet.

    This adds a new row to the table of parameters. Double-click each field in the row to provide a name, display name, and description for the parameter.

    Repeat this step to add more public parameters. When you are done, click **Next.**

    Public portlet parameters enable a portlet to communicate with the page on which it resides and with other portlets on that page. For more information, see Section 61.3.3, "How to Implement Public Parameters."

Figure 60–16 The Public Portlet Parameters Page



21. Oracle PDK-Java events are not supported in Framework applications, so On the Public Portlets Events page, click **Finish.**

    For more information about events, see the *Oracle Fusion Middleware Developer's Guide for Oracle Portal*.

60.2.5 What Happens When You Create a PDK-Java Portlet

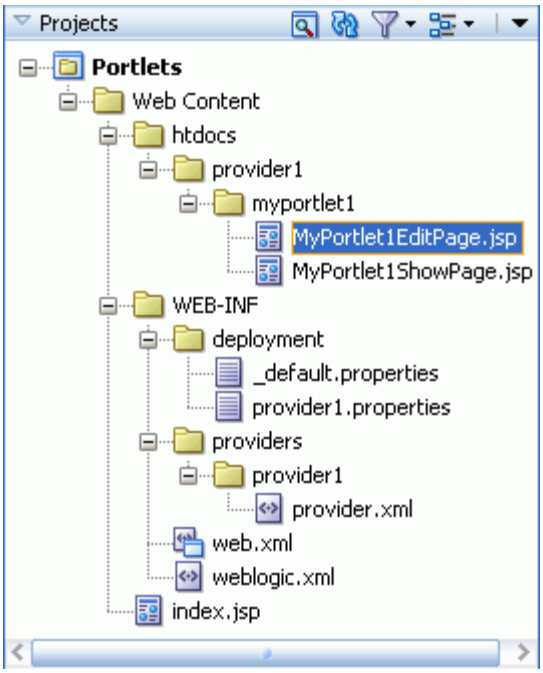
When you use the JDeveloper wizard to create a PDK-Java portlet, JDeveloper generates a default implementation of the portlet. Specifically, the following files are created:

- Files for each portlet mode you selected, for example *portletname*`EditPage.jsp`.
- `provider.xml` is the producer definition file that contains details of the portlets belonging to the producer.
- `web.xml` is the web deployment file for the application.
- `weblogic.xml` includes a shared library definition pointing to the PDK-Java shared library.
- `index.jsp` is used by JDeveloper for testing purposes.
- `_default.properties` is the default properties file.
- *`serviceID`*`.properties` is the properties file for the producer identified by *serviceID*.

All these files are required to deploy and run the portlet successfully, except for `index.jsp`.

You can see all these files in the Application Navigator, as shown in Figure 60–17.

Figure 60–17 Files Generated for a PDK-Java Portlet



60.2.6 After You Create an Oracle PDK-Java Portlet Using the JDeveloper Wizard

The next step is to extend the sample code with your own business logic to implement the desired functionality and features for your portlet.

For more information about the PDK-Java, see the *Oracle Fusion Middleware Java API Reference for Oracle PDK Java*.

60.3 Troubleshooting Java Portlets

This section provides information to assist you in troubleshooting problems you may encounter while creating portlets using the Create Portlet Wizards.

60.3.1 Cannot Access the Create Portlet Wizard

Problem

In the New Gallery, I cannot find the Oracle PDK-Java Portlet or Standards-based Java Portlet (JSR 286) options.

Cause

The application in which you are trying to create the JSR 286 or PDK-Java portlet was created using WebCenter Portal's Framework application template and therefore is not scoped for portlet creation.

Solution

1. Try creating the portlet in a Portlet Producer Application or any application scoped for portlet creation (any application except for those built using WebCenter Portal's Framework application template).
2. In the New Gallery, click the All Technologies tab to list all available options regardless of the technology scope of the application.

60.3.2 Cannot Add the Portlet Functionality that I Want to the Portlet

Problem

I cannot find the option to add certain features, for example portlet events or public render parameters, to the portlet in the Create Portlet Wizard.

Cause

The Create Portlet Wizard does not provide the option to add certain features to portlets.

Solution

After you create the portlet using the Create Portlet Wizard, edit the `portlet.xml` file using the Overview Editor to add advanced functionality. For more information, see Section 61.2, "Enhancing JSR 286 Java Portlets."

61 Coding Portlets

This chapter explains how you can enhance the Java portlets you created with the Oracle JDeveloper Create Java Portlet wizards.

This chapter includes the following sections:

- Section 61.1, "General Guidelines for Coding Java Portlets"
- Section 61.2, "Enhancing JSR 286 Java Portlets"
- Section 61.3, "Enhancing PDK-Java Portlets"
- Section 61.4, "Testing Portlet Personalization"
- Section 61.5, "Creating a Struts Portlet"

Before You Begin

Before you begin looking through this chapter ensure that:

- You are familiar with portlet terminology such as portlet modes. For more information, see Chapter 58, "Overview of Portlets."
- You have access to a WebLogic Server (WLS) managed server that is configured for use as a portlet container, such as the WLS_Portlet server.
- You are familiar with JDeveloper and know how to build and deploy Java components using it. You can download JDeveloper from OTN:

  `http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html`

61.1 General Guidelines for Coding Java Portlets

When you write your portlets in Java for either JSR 286 or PDK-Java, you should follow the best practices described in this section.

This section includes the following subsections:

- Section 61.1.1, "Guidelines for Portlet Modes"
- Section 61.1.2, "Guidelines for Navigation within a Portlet"
- Section 61.1.3, "Guidelines for JavaScript"

61.1.1 Guidelines for Portlet Modes

Portlet modes exhibit the runtime portlet functionality seen by users. You can provide an extended set of modes in addition to the standard ones provided with WebCenter Portal. Different portal products support extended portlet modes as follows:

- WebCenter Portal has defined an extended set of portlet modes. Different modes are available for both PDK-Java and JSR 286 portlets. For example, the JSR 286 Java Portlet wizard for JSR 286 portlets includes an extended mode called print, which you can use to provide a printer friendly version of the portlet. PDK-Java offers some modes not offered by JSR 286. If you are coding portlets to JSR 286, then you can declare custom portlet modes in the `portlet.xml` file that map to the extended modes offered by PDK-Java, or to accommodate any other functionality you may want to provide.

  Defining custom modes is especially useful if the portlet must interoperate with portal implementations from other vendors. You can offer any of these modes to users. In fact, it is recommended that some modes like Edit Defaults are offered.

- Third party portal products may have their own set of extended modes (JSR 286 custom modes) that producers can offer. In WebCenter Portal: Framework applications, the portlet chrome shows only the custom modes that are defined in WebCenter Portal: Framework. Arbitrary custom modes that a third party or custom portlet producer offers are ignored and therefore not supported.

This section includes the following subsections that discuss each of the standard portlet modes:

- Section 61.1.1.1, "Guidelines for View Mode"
- Section 61.1.1.2, "Guidelines for Edit Mode"
- Section 61.1.1.3, "Guidelines for Edit Defaults Mode"
- Section 61.1.1.4, "Guidelines for Help Mode"
- Section 61.1.1.5, "Guidelines for About Mode"
- Section 61.1.1.6, "Guidelines for Preview Mode"
- Section 61.1.1.7, "Guidelines for Full Screen Mode (PDK-Java Only)"

61.1.1.1 Guidelines for View Mode

View mode is known as Shared Screen mode in PDK-Java. For information about View mode, see Section 58.1.3.1, "View Mode."

When developing portlets, you must consider all of the factors that may influence the portlet's appearance on the page, such as the portlet's containing object and the other portlets with which your portlet shares the page. For example, suppose you choose to place your portlet inside an HTML table cell. The portlet should display only content that can be rendered within a table cell. Furthermore, the actual size of the table cell may vary depending on user settings, the browser width, and the amount and style of content in the portlet.

61.1.1.1.1 HTML Guidelines for Rendering Portlets Plain HTML is the most basic way to render portlets and provides a great deal of flexibility to portlet developers. You can use almost any standard HTML paradigm, such as links, forms, images, tables, if it can display within an HTML table cell. Improperly written HTML may appear inconsistently across different browsers and, in the worst case, could cause parts of your page not to appear at all. Ensure that you adhere to the following rules:

- **Use standard HTML:** The official HTML specification is available from the W3C (more information available at: `http://www.w3.org/`).
- **Avoid unterminated and extraneous tags:** The behavior of pages with improperly terminated tags is unpredictable because it depends on what the browser chooses

to do. Tools like weblint and HTML Tidy can help detect and fix hanging and unnecessary tags.

- **Consider restrictions imposed by container objects:** If your portlet is contained inside of an HTML element, such as a table cell, then you must ensure that your portlet can be rendered within that container. For example, if you place a portlet in a table cell, then you could not use frames in the portlet because they do not appear when inserted in a table.
- **Keep portlet content concise:** Do not try to take full screen content and expose it through a small portlet. You may end up with portlet content too small or cramped for smaller monitors. Full screen content is best viewed in Full Screen mode of PDK-Java.
- **Do not create fixed-width HTML tables in portlets:** You have no way to tell how wide a column your portlet has on a user's page. If your portlet requires more room than given, then it might overlap with another portlet in certain browsers.
- **Avoid long, unbroken lines of text:** The result is similar to what happens with wide fixed-width tables. Your portlet might overlap other portlets in certain browsers.
- **Check behavior when resizing the page:** Test your portlet's behavior when the browser window is resized to ensure that it works in different browser window sizes.
- **Check behavior when the default browser font changes:** People may choose whatever font size they want and they can change it at any time. Your portlet should handle these situations gracefully.

The HTML you use also affects the perceived performance of your site. Users judge performance based on how long it takes for them to see the page they requested, and browsers require time to interpret and display HTML. Given that, you should consider the following:

- **Avoid lengthy, complex HTML:** Portlets share a page with other portlets. Thus, portlet generation times can significantly effect the overall performance of the page. If portlets must render complex HTML or wait for external resources, such as third-party applications, then it can greatly slow the rendering of the page.

61.1.1.1.2 Cascading Style Sheet Guidelines for Rendering Portlets

The fonts and colors of every portlet on a page should match the style settings chosen by the user. To accomplish this goal, these style selections are embedded automatically using a Cascading Style Sheet (CSS) on each page. The portlets access these settings for their fonts and colors, either directly or using the Application Programming Interface (API).

While different browsers have implemented varying levels of the full CSS specification, WebCenter Portal: Framework uses a very basic subset of this specification to allow for consistent fonts and colors. CSS implementation levels should not affect the consistency of your pages across browsers. Follow these guidelines for using CSS:

- **Use CSS instead of hard coding:** Hard coding fonts and colors is extremely dangerous. If you hard code fonts and colors, then your portlet may look out of place when the user changes the page style settings. Since you have no way of knowing the user's font and color preference choices, you might also choose to hard code a font color that turns out to be the same as the user's chosen background color, in which case your portlet appears to be invisible to that user.
- **Use the CSS APIs to format your text:** The style sheet definition is available at the top of pages, but you should not call it directly. Instead, use the APIs provided to

format your text appropriately. This method ensures that your portlets work even if the style sheet changes in the future.

- **Avoid using CSS for absolute positioning:** Since users can personalize their pages, you cannot guarantee that your portlet can appear in a particular spot.
- **Follow Accessibility Standards:** You should ensure that you code your style sheets according to existing accessibility standards (more information available at `http://www.w3.org/TR/WCAG10-CSS-TECHS/`).

61.1.1.2 Guidelines for Edit Mode

For information about Edit mode, see Section 58.1.3.2, "Edit Mode."

61.1.1.2.1 Guidelines for Edit Mode Operations The following guidelines should govern what you expose to users in Edit mode:

- **Enable users to personalize the title of the portlet:** The same portlet may be added to the same page several times. Enabling the user to personalize the title helps alleviate confusion.
- **If using caching, invalidate the content:** If personalizations cause a change in portlet display or content, then you must ensure that the portlet content is regenerated and not returned from the cache. Otherwise, the user may see incorrect content.
- **Do not use Edit mode as an administrative tool:** Edit mode is meant to give users a way of changing the behavior of their portlets. If you want to change producer settings or do other administrative tasks, then you should create secured portlets specifically for those tasks.

61.1.1.2.2 Guidelines for Buttons in Edit Mode For consistency and user convenience, Edit mode should implement the following buttons in the following order:

- **OK** saves the user personalizations and returns the portlet to view mode.
- **Apply** saves the user personalizations and reloads the current page.
- **Cancel** returns the portlet to view mode without saving changes.

61.1.1.2.3 Guidelines for Rendering Personalization Values When you show the forms used to change personalization settings, you should default the values such that the user does not have to constantly reenter settings. When rendering the personalization values, use the following sequence to provide consistent behavior:

1. **User preference:** Query and display this user's personalizations, if available.
2. **Instance defaults:** If no user personalizations are found, then query and display system defaults for the portlet instance. These are set in Edit Defaults mode and apply only to this portlet instance.
3. **Portlet defaults:** If no system default personalizations are found, then display general portlet defaults, which may be blank. General portlet defaults are sometimes hard coded into the portlet but should be overridden if either of the two previous conditions apply.

This logic enables the personalizations to be presented in a predictable way, consistent with the other portlets in the Framework application.

61.1.1.3 Guidelines for Edit Defaults Mode

For information about Edit Defaults mode, see Section 58.1.3.3, "Edit Defaults Mode."

61.1.1.3.1 Guideline for Edit Default Mode Options The following guideline should govern what you expose to page designers in Edit Defaults mode:

- **Do not use Edit Defaults mode as an administrative tool.** Edit Defaults mode gives users a way of changing the behavior of their portlets. If you want to change producer settings or do other administrative tasks, then you should create secured portlets specifically for those tasks.

61.1.1.3.2 Guidelines for Buttons in Edit Defaults Mode For consistency and user convenience, Edit Defaults mode should implement the following buttons in the following order:

- **OK** saves the user personalizations and returns the portlet to view mode.
- **Apply** saves the user personalizations and reloads the current page.
- **Cancel** returns the portlet to view mode without saving changes.

61.1.1.3.3 Guidelines for Rendering Personalization Values When you show the forms used to change personalization settings, you should default the values so that the application developer does not have to constantly reenter settings. When rendering personalization values, use the following sequence to provide consistent behavior:

1. **Instance preferences:** Query and display system defaults for the portlet instance.
2. **Portlet defaults**: If no system default personalizations are found, then display general portlet defaults, which may be blank. General portlet defaults are sometimes hard coded into the portlet but should be overridden by system defaults.

This logic enables the personalizations to be presented in a predictable way, consistent with the other portlets in the Framework application.

61.1.1.4 Guidelines for Help Mode

For information about Help mode, see Section 58.1.3.4, "Help Mode."

The following guideline should govern what you expose to users in Help mode:

- **Describe how to use the portlet.** Users may not know all the features your portlet provides just from its interface. Describe the features and how to get the most out of them.

61.1.1.5 Guidelines for About Mode

For information about About mode, see Section 58.1.3.5, "About Mode."

The following guideline should govern what you expose to users in About mode:

- **Display relevant copyright, version, and author information.** Users want to know what portlet they are using and where they can get more information. The about page may become important when supporting your portlets.

61.1.1.6 Guidelines for Preview Mode

> **Note:** This mode has no particular application in Framework applications, but used in Oracle Portal's Portlet Repository.

The following guidelines should govern what you expose to users in Preview mode:

- **Provide an idea of what the portlet does.** Preview mode should generate enough content for the user to get an idea of the actual content and functionality of the portlet.
- **Keep your portlet previews small.** The amount of data produced in this mode should not exceed a few lines of HTML or a screen shot. Preview mode appears in a small area, and exceeding the window's size looks unprofessional and forces users to scroll.
- **Do not use live hyperlinks.** Links may not work as expected when rendered in Preview mode. Hyperlinks can be simulated using the underline font.
- **Do not use active form buttons.** Forms may not work as you expect them to when rendered in Preview mode. If you decide to render form elements, then do not link them to anything.

61.1.1.7 Guidelines for Full Screen Mode (PDK-Java Only)

Technically, JSR 286 portlets do not have Full Screen mode. However, you can implement the equivalent of Full Screen mode for a JSR 286 portlet with View mode and a maximized state for the window.

61.1.2 Guidelines for Navigation within a Portlet

In some ways, navigation between different sections or pages of a single portlet is identical to navigation between standard web pages. Users can submit forms and click links. In typical, simple web pages, both of these actions involve sending a message directly to the server responsible for rendering the new content, which is then returned to the client. In portlets, which comprise only part of a page, the form submission or link rendered within the portlet does not directly target the portlet. It passes information to the portlet through the Framework application. If a link or form within a portlet does not refer back to the application, then following that link takes the user away from the application, which is not typically the desired behavior.

The component developer does not need to know the detailed mechanics of how the parameters of a form or link get passed around between the user, application, and portlet. However, they must understand that they cannot write links in a portlet the same way they do for typical, simple web pages.

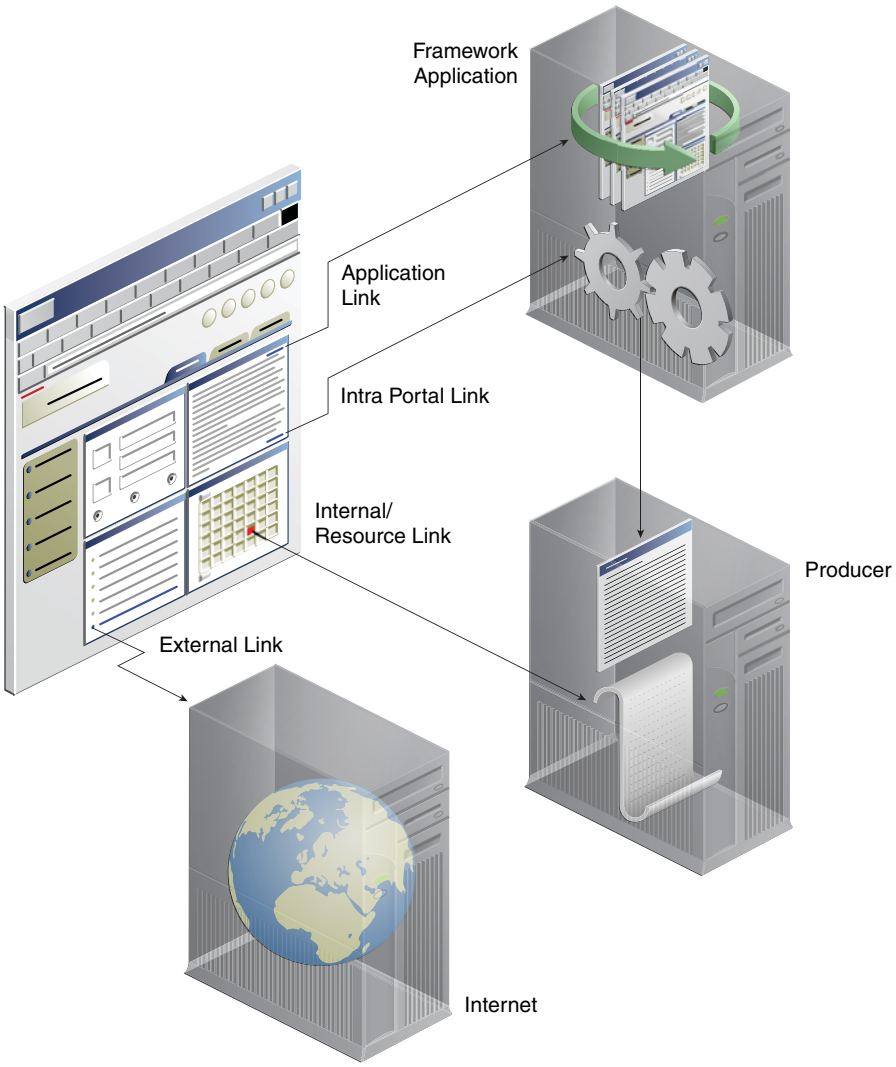
Types of Links for Portlets

A portlet may render links of four classes, as follows:

- **Intraportlet links** require the portlet to be aware of the address of the Framework application because they actually refer to it in some way.
- **Application links**, like intraportlet links, must be aware of the address of the Framework application for the same reason.
- **External links** make no reference to the Framework application and work in portlets as they would do in a normal web page.
- **Internal/Resource links**, like external links, also make no reference to the Framework application.

Figure 61–1 contains a summary of these link types. The arrows indicate how the links reference the resources to which they logically refer.

Figure 61–1 WebCenter Portal: Framework Application Link Types



This section includes the following subsections:

- Section 61.1.2.1, "Intraportlet Links"
- Section 61.1.2.2, "Application Links"
- Section 61.1.2.3, "External Links"
- Section 61.1.2.4, "Internal/Resource Links"

61.1.2.1 Intraportlet Links

Intraportlet links go to different sections or pages within a given portlet. Strictly speaking, they refer to the page containing the portlet, but they contain parameters that cause the portlet to render a different section or page within that page when it is requested by the user.

As a direct consequence, a portlet cannot expect to render links to different sections or pages of itself using relative links or absolute links based on its own server context. Intraportlet link are useful for intraportlet navigation, either as links or form submission targets.

61.1.2.2 Application Links

Application links refer to significant pages within the Framework application, such as the user's home page.

61.1.2.3 External Links

External links refer neither to the portlet (through a page) nor to any part of the Framework application. If selected, these links take the user away from the application, for example, `www.oracle.com`.

61.1.2.4 Internal/Resource Links

Internal/Resource links refer to internal (to the portlet) resources. Sometimes they are exclusively used internally during portlet rendering, for example as a server side include. On other occasions, they may be used externally to reference portlet resources like images. In this latter case, you can use the PDK-Java `constructResourceURL` method in the `UrlUtils` class to retrieve images from behind a firewall using resource proxy. Note that in order for resource proxying to work, you must first set the JNDI variable, `oracle/portal/provider/sample/resourceUrlKey`, for the producer. For more information about setting JNDI variables, see Section 61.3.5.2, "Setting JNDI Variable Values."

For example, `lottery.jsp` of the lottery sample, which is available with PDK-Java, contains resource proxy requests for images.

```
<%@ page contentType="text/html;charset=UTF-8" %>
<%@ page session="false" import="oracle.portal.provider.v2.render.*" %>
<%@ page import="oracle.portal.provider.v2.render.http.HttpPortletRendererUtil" %>
<%@ page import="oracle.portal.provider.v2.url.UrlUtils" %>
<%@ page import="oracle.portal.sample.v2.devguide.lottery.*" %>
<%
  LottoPicker picker = new LottoPicker();
  picker.setIdentity(request.getRemoteAddr() ); %>
<% PortletRenderRequest portletRequest = (PortletRenderRequest)
request.getAttribute("oracle.portal.PortletRenderRequest"); %>
<% String name = portletRequest.getUser().getName(); %>
<P class="PortletHeading1" ALIGN="CENTER">Hi <%= name %>, Your Specially
  Picked</P>
<P ALIGN="CENTER"><IMG SRC="<%= UrlUtils.constructResourceURL(portletRequest,
  HttpPortletRendererUtil.absoluteLink(request, "images/winningnumbers.gif")) %>"
  WIDTH="450" HEIGHT="69" ALIGN="BOTTOM" BORDER="0"></P>
<P>
<P ALIGN="CENTER">
<TABLE ALIGN="CENTER" BORDER="0" CELLPADDING="0" CELLSPACING="0">
<TR>
<%
     int [] picks = picker.getPicks();
     for (int i = 0; i < picks.length; i++) {
%>
             <TD>
    <IMG SRC="<%= UrlUtils.constructResourceURL(portletRequest,
      HttpPortletRendererUtil.absoluteLink(request, "images/ball" + picks[i])) %>
      .gif" WIDTH="68" HEIGHT="76" ALIGN="BOTTOM" BORDER="0">
     </TD>
<%
     }
```

For session-based producers, any cookies returned from the original `initSession` call to the producer are sent with the request back to the producer to maintain the right session context.

61.1.3 Guidelines for JavaScript

JavaScript can often be useful within a portlet, but bear in mind the following guidelines within your portlets:

- You should never use JavaScript to redirect the page in which the portlet is rendered. If you want to direct users elsewhere, then you should do so in your portlet action handling code or open a new window in the browser.
- Ensure that identifiers in your JavaScript are qualified. By qualifying your identifiers, you ensure that they are unique and do not clash with any JavaScript on the page.

61.2 Enhancing JSR 286 Java Portlets

When you have built your initial portlet in the Create JSR 286 Java Portlet wizard as described in Section 60.2.1, "How to Create a JSR 286 Java Portlet," the next step is to enhance it. Because JSR 286 portlets adhere to the Java standards, you can find substantial information about enhancing them from many different sources, such as third-party books and web pages.

This section includes the following subsections:

- Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File"
- Section 61.2.2, "How to Add Custom Portlet Modes"
- Section 61.2.3, "How to Access User Information"
- Section 61.2.4, "How to Customize the Runtime Environment"
- Section 61.2.5, "How to Use Portlet Events"
- Section 61.2.6, "How to Use Public Render Parameters"
- Section 61.2.7, "How to Add Personalization"
- Section 61.2.8, "How to Use Portlet Filters"
- Section 61.2.9, "How to Implement Portlet Caching"
- Section 61.2.10, "How to Implement Export of Customizations (WSRP 2.0)"
- Section 61.2.11, "How to Implement Rewritten URLs for Resource Proxy"
- Section 61.2.12, "How to Implement Stateless Resource Proxying"
- Section 61.2.13, "How to Enable Java Object Cache for Preference Store Access"
- Section 61.2.14, "How to Implement Security for JSR 286 Portlets"

61.2.1 How to Edit the Portlet Deployment Descriptor File

The portlet deployment descriptor file for the application, `portlet.xml`, specifies the portlet resources in the application. After you have created your JSR 286 portlet using the Portlet Wizard, you can edit the portlet.xml file to edit those resources.

When you open the portlet.xml file in JDeveloper, you can edit the source code or you can use the Overview Editor to edit portlet resources without having to manually modify it in Source mode.

To edit the portlet deployment file:

1. In the JDeveloper Application Navigator, open the portlet producer application.
2. Expand the portlet project.
3. Expand the **Web Content** node and then the **WEB-INF** node.
4. Right-click **portlet.xml** and choose **Open**.
5. Click the **Design** tab to open the Overview Editor for `portlet.xml`.
6. The Overview Editor for portlet.xml contains the following tabs:
   - **Application**—Use to specify general information for the portlet producer application. These properties apply to all portlets within the application.
   - **Portlets**—Use to specify information for individual portlets within the application. For example, you can specify the portlet events and public render parameters supported by a portlet.
   - **Events**—Use to create and manage portlet events for use by all portlets in the application. For more information, see Section 61.2.5, "How to Use Portlet Events."
   - **Parameters**—Use to create and manage public render parameters for use by all portlets in the application. For more information, see Section 61.2.6, "How to Use Public Render Parameters."
   - **Filters**—Use to specify filter information for the portlets in the application. For more information, see Section 61.2.8, "How to Use Portlet Filters."
   - **Filter Mappings**—Use to assign portlet filters to individual portlets. For more information, see Section 61.2.8, "How to Use Portlet Filters."

61.2.2 How to Add Custom Portlet Modes

In the Create JSR 286 Java Portlet wizard, you add portlet modes by adding them to a list on the Content Types and Portlet Modes page. For more information about using the wizard, see Section 60.2.1, "How to Create a JSR 286 Java Portlet." The wizard enables you to implement the standard modes supported by JSR 286 and the extended modes provided by WebCenter Portal.

The standard modes supported by JSR 286 are:

- View
- Edit
- Help

WebCenter Portal supports the following additional custom modes:

- about
- config
- edit_defaults
- preview
- print

After the initial creation of the portlet, you can also define your own custom portlet modes. You can add custom modes to map to the extra modes offered by PDK-Java (for example, Full Screen mode), or to accommodate any other unique functionality you may want to provide.

The principles of implementing portlet modes are the same for all modes.

Before You Begin

The steps that follow assume that you have:

- Built a portlet using the Create JSR 286 Java Portlet wizard
- Successfully registered the portlet's producer
- Added the portlet to a page

To add a custom portlet mode:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Application** tab, if necessary.
3. Expand the Custom Modes section, if necessary.

   This section lists all the custom modes available to the portlets within the application. It includes the WebCenter Portal extended portlet modes.
4. Click the **Add Custom Mode** icon to create a new row for the portlet mode.
5. In the **Portlet Mode** field, enter the name of the portlet mode.

   The name must be unique within the application.
6. In the **Description** field, enter a brief description to explain the purpose of the mode.
7. Deselect the **Portal Managed** checkbox if the portlet mode is not managed by the portal
8. Save the `portlet.xml` file.
9. After adding the custom portlet mode to the application, you must then create the code for the mode in the portlet implementation file.

61.2.3 How to Access User Information

You can create user attributes in the Portlet Producer application to access commonly used user information, such as `user.login.id` or `user.name.family`. At runtime, these user attributes are mapped to the actual attributes of the current user. Portlets can use this information to obtain information about the current user.

To access user information:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Application** tab, if necessary.
3. Expand the User Attributes section, if necessary.
4. Click the **Add User Attribute** icon to create a new row for the attribute.
5. In the **Name** field, enter the name of the user attribute, for example `user.login.id`.
6. In the **Description** field, enter a description for the user attribute.

   At runtime, the portlet container maps this attribute to the appropriate value. For example, if the current user is John Doe, then the `user.name.family` user

attribute is mapped to Doe. Portlets can obtain a `Map` object containing the user attributes from the `PortletRequest` interface.

7. Save the `portlet.xml` file.

61.2.4 How to Customize the Runtime Environment

When a portlet is run, the portlet container provides the runtime environment and provides an interface between the Portlet Producer application and the portlet.

Container runtime options provide a way to customize the behavior of the portlet container and therefore customize the runtime environment.

This section includes the following subsections:

- Section 61.2.4.1, "Supported Container Runtime Options"
- Section 61.2.4.2, "Setting Container Runtime Options for All Portlets in an Application"
- Section 61.2.4.3, "Setting Container Runtime Options for Individual Portlets"

61.2.4.1 Supported Container Runtime Options

Table 61–1 lists the container runtime options supported by the WebCenter Portal portlet container.

For more information about the JSR 286 container runtime options, see the JSR 286 specification at:

`http://jcp.org/en/jsr/detail?id=286`

Table 61–1 Supported Container Runtime Options

Container Runtime Option	Supported by	JSR 286 Standard
javax.portlet.actionScopedRequestAttributes	Application Portlet	Yes
javax.portlet.escapeXml	Application Portlet	Yes
java.portlet.renderHeaders	Not supported	Yes
javax.portlet.servletDefaultSessionScope	Application Portlet	Yes
com.oracle.portlet.allowEventPayloadsWithoutJAXBBinding	Application	No
com.oracle.portlet.allowWsrpExport	Application	No
com.oracle.portlet.compatibilityMode	Application Portlet	No
com.oracle.portlet.defaultProxiedResourceRequiresWsrpRewrite	Application Portlet	No
com.oracle.portlet.defaultServedResourceRequiesWsrpRewrite	Application Portlet	No
com.oracle.portlet.disallowResourceServing	Application Portlet	No

Table 61–1 (Cont.) Supported Container Runtime Options

Container Runtime Option	Supported by	JSR 286 Standard
com.oracle.portlet.escapeXmlEncodeUrls	Application Portlet	No
com.oracle.portlet.eventPayloadsXmlType	Application	No
com.oracle.portlet.excludedActionScopeRequestAttributes	Application Portlet	No
com.oracle.portlet.externalScopeRequestAttributes	Application Portlet	No
com.oracle.portlet.importCssToIFrame	Application Portlet	No
com.oracle.portlet.minimumWsrpVersion	Application Portlet	No
com.oracle.portlet.offerPortletOverWsrp	Application Portlet	No
com.oracle.portlet.portalInfoProvider	Application Portlet	No
com.oracle.portlet.redirectAfterAction	Application Portlet	No
com.oracle.portlet.requireIFrame	Application Portlet	No
com.oracle.portlet.streamingOptimized	Application Portlet	No
com.oracle.portlet.suppressWsrpOptimisticRender	Application Portlet	No
com.oracle.portlet.trapWsrpRenderExceptions	Application Portlet	No
com.oracle.portlet.trimEncodeUrls	Application Portlet	No
com.oracle.portlet.useWsrpUserContextForUserAuthentication	Application	No
com.oracle.portlet.wsrpHeaderMode	Application Portlet	No
com.oracle.portlet.wsrpLegacyPortletHandle	Portlet	No
com.oracle.portlet.wsrpPortletHandle	Portlet	No
oracle.portlet.bridge.adf.raiseUndeclaredContextualEvents	Portlet	No

javax.portlet.actionScopedRequestAttributes

Specifies whether to store action-scoped request attributes so that they are available to portlets until a new action occurs.

You can use the WebCenter Portal-specific `excludedActionScopeRequestAttributes` container runtime option to limit which request attributes are stored in the scopes.

Valid values:

- `true`—store request attributes starting at `processAction` until the next `processAction`. You can specify a second value of `numberOfCachedScopes` and a third value indicating the number of scopes to be cached by the portlet container.
- `false`—do not store request attributes.

javax.portlet.escapeXml

Specifies whether to XML encode URLs returned from `actionUrl`, `renderUrl`, and `resourceUrl` JSR 286 tag library tags.

You can override this option on a per-tag basis using the `encodeXml` attribute.

Valid values:

- `true`—(default) the URLs returned by the `actionUrl`, `renderUrl`, and `resourceUrl` tags are XML encoded (using ampersand entities for parameter separation).
- `false`—the URLs returned by the actionUrl, renderUrl, and resourceUrl tabs are not XML encoded (and use just ampersand characters).

Note: This option does not have any effect on the return value of `PortletURL.toString()` or `ResourceURL.toString()`; in both cases, for JSR 286, the output uses only ampersand characters. To use ampersand entities or be able to specify the XML encoding to use when generating URLs not using the tag libraries, see the `BaseURL.write()` methods.

---

javax.portlet.servletDefaultSessionScope

Specifies the scope of the session object provided to servlets or JSPs that are included or forwarded from a Java portlet.

Valid values:

- `APPLICATION_SCOPE`—(default) map the portlet session with application scope.
- `PORTLET_SCOPE`—map the portlet session with portlet scope.

com.oracle.portlet.allowEventPayloadsWithoutJAXBBinding

Allows event payload types declared in `portlet.xml` and event payloads sent from JSR 286 portlets to bypass the JSR 286 spec requirement that these types have a valid JAXB binding.

Valid values:

- `true`—event payloads without valid JAXB bindings are allowed.
- `false`—event payloads without valid JAXB bindings are not allowed, per the JSR 286 specification.

com.oracle.portlet.allowWsrpExport

Specifies whether the WSRP export-portlets operation should be supported for the web application.

This option is used mainly for backward-compatibility; it is preferred to control the export-portlets operation through the `wsrp-producer-config.xml` setting.

Valid values:

- `true`—(default) export-portlets is allowed.
- `false`—export-portlets is not allowed. This overrides the setting in `WEB-INF/wsrp-producer-config.xml`.

com.oracle.portlet.compatibilityMode

Set to `owc168` to invoke the WebCenter Portal JSR 286 container JSR 168 compatibility mode.

This also automatically sets the `disallowResourceServing` runtime option to `true` if no other value for that option is specified.

com.oracle.portlet.defaultProxiedResourceRequiresWsrpRewrite

Specifies the default WSRP `requiresRewrite` flag to use when encoding URLs for resources not served by the portlet. This setting is used for all URLs returned by the `PortletResponse.encodeURL()` method, unless overridden by the presence of the `oracle.portlet.server.resourceRequiresRewriting` request attribute when the `PortletResponse.encodeURL()` method is called.

Valid values:

- `true`—(default) the `requiresRewrite` flag is set to `true`, indicating that the resource should be rewritten by the consumer.
- `false`—the `requiresRewrite` flag is set to `false`, indicating that the resource does not necessarily need to be rewritten by the consumer.

com.oracle.portlet.defaultServedResourceRequiesWsrpRewrite

Specifies the default WSRP `requiresRewrite` flag to use when generating resource URLs for portlet-served resources.

This setting is used for all resource URLs created by the portlet, unless overridden by the presence of the `resourceRequiresRewriting` request attribute when the resource URL methods `write()` or `toString()` are called. This setting is also used to specify the WSRP `requiresRewriting` flag on the served resource response, but can be overridden by the presence of the `resourceRequiresRewriting` request attribute when the portlet's `serveResource()` method returns.

Valid values:

- `true`—the `requiresRewrite` URL flag and `requiresRewriting` response flag are set to `true`, indicating that the resource should be rewritten by the consumer
- `false`—the `requiresRewrite` URL flag and `requiresRewriting` response flag are set to `false`, indicating that the resource does not necessarily need to be written by the consumer, although the consumer may choose to rewrite the URL.

If unspecified, the `requiresRewrite` URL flag is not given a value, and the `requiresRewriting` response flag for a `serveResource` operation is based on the MIME type of the response.

com.oracle.portlet.disallowResourceServing

Specifies whether portlets are allowed to serve resources. This is useful if JSR 168 portlets are run in the 286 container, as any JSR 168 portlet extending `javax.portlet.GenericPortlet` automatically inherits the JSR 286 functionality, which automatically forwards resource requests to a file in the web application named after the resource ID, creating a potential security problem. For the same security reason, JSR 286 portlets that do not serve resources are safest to disallow resource serving.

Valid values:

- `true`—portlets are not allowed to serve resources.
- `false`—portlets are allowed to serve resources.

com.oracle.portlet.escapeXmlEncodeUrls

Specifies whether the JSR 286 container should XML-encode URLs generated from the `PortletResponse.encodeURL()` method.

Valid values:

- `true`—URLs generated by the `PortletResponse.encodeURL()` method are XML-encoded.
- `false`—(default) URLs generated by `PortletResponse.encodeURL()` are not XML-encoded.

com.oracle.portlet.eventPayloadsXmlType

Specifies optional XML schema types for JAXB-bindable Java object event payloads if events are sent over WSRP. This is a multi-valued runtime option; each value consists of a Java class name and QName pair, separated by a colon (`:`). The QName should use the standard Java String representation of a QName (`{`*`namespace`*`}`*`localpart`*). For each value specified, the QName is used as the XML schema type to annotate WSRP event payloads of the specified Java object type after the object has been marshalled to XML. This may be useful when using events to communicate across portlets on multiple producers.

com.oracle.portlet.excludedActionScopeRequestAttributes

This is a multi-valued property with each value being a regular expression. Request attributes which match any of the regular expressions are not stored as action-scoped request attributes if the `javax.portlet.actionScopedRequestAttributes` container runtime option is used, in addition to any request parameters whose values match the regular expressions defined in the `com.oracle.portlet.externalScopeRequestAttributes` container runtime option.

com.oracle.portlet.externalScopeRequestAttributes

This is a multi-valued property with each value being a regular expression. Request attributes which match any of the regular expressions are considered outside of portlet scope, and are shared with the underlying portal request.

If the `javax.portlet.actionScopedRequestAttributes` option is used, any request attributes matching the regular expressions declared in `externalScopeRequestAttributes` are not stored in the action scope request attributes.

com.oracle.portlet.importCssToIFrame

Specifies to a portal consumer whether the CSS file should be imported to an IFRAME portlet.

Valid values:

- `false`—(default) nothing is done.
- `true`—the CSS file from the consumer is applied to an IFRAME portlet.

com.oracle.portlet.minimumWsrpVersion

Specifies minimum required WSRP version for the portlet to work. If the WSRP version being used is less than the value specified, the portlet will not be included in a WSRP `GetServiceDescription`, `GetPortletDescription`, `GetMarkup` and other WSRP responses.

Valid values:

- `1`
- `2`

com.oracle.portlet.offerPortletOverWsrp

Specifies whether the portlet is offered in the WSRP producer's service description.

Any `offerRemote` setting in a `.portlet` file referencing the JSR 168/286 portlet overrides this container runtime option.

Valid values:

- `true`—(default) the portlet is offered in the WSRP producer's service description.
- `false`—the portlet is not included in the service description.

If not specified at all, the default value specified in the `WEB-INF/producer-config.xml` is used.

com.oracle.portlet.portalInfoProvider

Specifies the class name of an optional implementation of `com.bea.portlet.container.IPortalInfo` interface. This implementation defines the value returned by the JSR 286 container's `request.getPortalContext().getPortalInfo()`. The default implementation returns the producer/local server name and version, but a custom implementation could pass both the consumer server information and the producer server information in a WSRP scenario.

com.oracle.portlet.redirectAfterAction

Causes the JSR 286 container to send a redirect to the browser to the portlet's render URL after a `processAction` is run (and after events are handled) so that reloading the resulting page will not result in another `processAction`.

Valid values:

- `true`—a redirect is issued after every portlet action.
- `false`—no redirect is not automatically issued after portlet actions.

com.oracle.portlet.requireIFrame

Specifies whether the portlet must be rendered inside an IFRAME.

Valid values:

- `true`—renders the portlet inside an IFRAME.
- `false`—does not force the portlet to be rendered inside an IFRAME.

com.oracle.portlet.streamingOptimized

Indicates the portlet is optimized to run in streaming mode, which may enhance performance.

Valid values:

- `true`—the portlet is optimized to run in streaming mode.
- `false`—the portlet is not optimized to run in streaming mode.

com.oracle.portlet.suppressWsrpOptimisticRender

Suppresses the optimistic render of a portlet after the action and/or event lifecycles if the portlet is being run over WSRP.

Valid values:

- `true`—optimistic render is always suppressed.
- `false`—optimistic render may be performed.

com.oracle.portlet.trapWsrpRenderExceptions

Specifies whether the JSR 286 container should send exceptions during render as WSRP SOAP faults, or render an exception message for the portlet markup instead.

Valid values:

- `true`—(default) exceptions are not sent as SOAP faults but instead as rendered exception stack traces.
- `false`—exceptions generated by the portlet during render are treated as SOAP faults.

com.oracle.portlet.trimEncodeUrls

Specifies whether the JSR 286 container should trim whitespace from URLs passed to the `PortletResponse.encodeURL()` method.

Valid values:

- `true`—(default) leading and trailing whitespace is trimmed from the URL passed into `PortletResponse.encodeURL()`.
- `false`—whitespace is not trimmed from the URL passed into `PortletResponse.encodeURL()`.

com.oracle.portlet.useWsrpUserContextForUserAuthentication

Specifies whether the PortletRequest methods `getRemoteUser()`, `getUserPrincipal()` and `isUserInRole()` are based on the WSRP user context information or on standard J2EE security.

Valid values:

- `true`—the user information is based on the WSRP user context, if the portlet is run over WSRP. This can be a security problem so use this option with care.
- `false`—(or the portlet is not being run over WSRP) the user information is based on the J2EE authenticated user.

com.oracle.portlet.wsrpHeaderMode

Used only when portlets are being rendered as WSRP remote portlets, to indicate where cookies and headers should be put in the WSRP SOAP response as a hint to the WSRP consumer for the header or cookie's intended final destination.

When portlets are run locally (not over WSRP), headers and cookies set by portlets are always assumed to go to the client. Setting this container runtime option sets a default value for the `PortletRequest` attribute `com.oracle.portlet.wsrpHeaderMode`, which can still be overridden by the portlet at runtime on a per-header basis.

Valid values:

- `client`—headers and cookies set by the portlet are directed to go to the client (for example, the browser).
- `consumer`—headers and cookies set by the portlet are directed to go to the consumer and not passed on to the client.
- `both`—headers and cookies set by the portlet are directed to go to the client and the consumer.

If not set, the portlet container assumes the default value for the producer as specified in the `WEB-INF/wsrp-producer-config.xml` file.

com.oracle.portlet.wsrpLegacyPortletHandle

Allows the specification of a legacy WSRP portlet handle to be used for the portlet, which must be unique within the web application. This is useful for backward-compatibility with WebCenter Portal consumers that use the legacy, portlet-position-based WSRP portlet handles.

If specified, the legacy portlet handle is published in the WSRP serviceDescription as a legacy-handle extension on the portlet, and consumers are able to access the portlet using the legacy portlet handle, although the portlet will not be published in the WSRP serviceDescription under the legacy portlet handle.

com.oracle.portlet.wsrpPortletHandle

Allows the specification of the WSRP portlet handle to be used for the portlet, which must be unique within the web application.

oracle.portlet.bridge.adf.raiseUndeclaredContextualEvents

Allows the Oracle JSF Portlet Bridge to raise ADFm events that do not have corresponding portlet events declared in the `portlet.xml` file.

Valid values:

- `true`—any ADFm event raised is forwarded on as a portlet event.
- `false`—only events with corresponding portlet event declarations are forwarded.

61.2.4.2 Setting Container Runtime Options for All Portlets in an Application

Setting container runtime options at the application level affects all the portlets in the application.

To set application-level container runtime options:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Application** tab, if necessary.

3. Expand the Container Runtime Options section, if necessary.
4. Click the **Add Container Runtime Option** icon to display a list of available options.

   The list includes all container runtime options supported by the WebCenter Portal portlet container, which are listed in Table 61–1.

   If you are using a different portlet container that supports additional container runtime options that are not listed, select **<Customize>** and enter the name of the option in the **Name** field.

   If you specify a container runtime option that is not supported by the portlet container, it is ignored.
5. In the **Value** field, enter a value for the container runtime option.

   The container runtime option is added to the `portlet.xml` code.

```
<container-runtime-option>
   <name>com.oracle.portlet.requireIFrame</name>
   <value>true</value>
</containter-runtime-option>
```

6. Save the `portlet.xml` file.

61.2.4.3 Setting Container Runtime Options for Individual Portlets

You can set container runtime options at the individual portlet level to override the application-wide settings.

To set portlet-level container runtime options:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Portlets** tab, if necessary.
3. Click the **Advanced** tab.
4. Click the **Add Container Runtime Option** icon to display a list of available options.

   The list includes all container runtime options supported by the WebCenter Portal portlet container, which are listed in Table 61–1.

   If you are using a different portlet container that supports additional container runtime options that are not listed, select **<Customize>** and enter the name of the option in the **Name** field.

   If you specify a container runtime option that is not supported by the portlet container, it is ignored.
5. In the **Value** field, enter a value for the container runtime option.

   The container runtime option is added to the `portlet.xml` code.

```
<container-runtime-option>
   <name>com.oracle.portlet.requireIFrame</name>
   <value>false</value>
</containter-runtime-option>
```

6. Save the `portlet.xml` file.

61.2.5 How to Use Portlet Events

Portlet events are a JSR 286 feature that enable inter-portlet communication by providing portlets with the ability to respond to actions that occur outside of the portlet itself, for example an action performed on the page that contains the portlet or on another portlet on the same page. Portlet events can be cascaded so that a portlet may respond to an event by triggering an event of its own, which in turn affects other portlets on the page.

Any portlet events supported by a portlet must be declared in the application section of the portlet deployment descriptor (`portlet.xml`). Portlet events defined at the application level in this way are available to all the portlets in the application.

Individual portlets within the application can then specify which of these portlet events they want to use. Portlets can declare events that they are interested in receiving, called *processing events*, and events that they trigger, called *publishing events*.

This section includes the following subsections:

- Section 61.2.5.1, "Defining a Portlet Event at the Application Level"
- Section 61.2.5.2, "Adding a Processing Event to a Portlet"
- Section 61.2.5.3, "Adding a Publishing Event to a Portlet"

For more information about portlet events and how to implement them, see the JSR 286 specification at:

`http://jcp.org/en/jsr/detail?id=286`

61.2.5.1 Defining a Portlet Event at the Application Level

For a portlet event to be available to a portlet, it must first be declared in the application section of the portlet deployment descriptor (`portlet.xml`).

To define a portlet event at the application level:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Events** tab.
3. Click the **Add** icon to create a new portlet event.
4. From the dropdown list, select whether you are specifying a fully qualified name (QName) for the portlet event, or just the local part of the name.

   A QName uniquely identifies the portlet event across applications by specifying a namespace for the parameter as well as a local name. A page typically contains multiple portlets which may come from different applications. Using QNames ensures that portlet events from one portlet do not unintentionally interfere with the other portlets on a page regardless of where those portlets come from.

   If the namespace for the portlet event is the same as the application default namespace, you can omit the namespace when defining the event by specifying an unqualified name. If the application default namespace has not been defined, a portlet event with an unqualified name uses the XML default namespace.
5. If you selected **Qualified Name**:
   a. In the **Namespace** field, enter the namespace for the portlet event.
   b. In the **Name** field, enter the local part of the portlet event name.
6. If you selected **Unqualified Name**, in the **Name** field, enter the local part of the portlet event name. The event uses the application default namespace as the

namespace for the event (or the XML default namespace if no namespace is defined for the application).

7. In the **Payload** field, enter or browse for the data type of the payload provided by the portlet event.
8. In the **Description** field, enter a description for the portlet event.
9. (Optional) In the Aliases panel, you can provide a list of aliases so that a portlet event can be recognized by the portlet even if it has a different QName from the one defined by the portlet.
   a. Click the **Create** icon to create a new alias for the portlet event.
   b. In the **Namespace** field, enter the namespace of the portlet event to use for the alias.
   c. In the **Name** field, enter the local part of the portlet event to use for the alias.
10. The portlet event is added to the application definition section of the `portlet.xml` file:

```
<event-definition>
   <description>This is my event</description>
   <qname xmlns:x="http://example.com/events">myEvent</qname>
   <value-type>java.lang.String</value-type>
</event-definition>
```

11. Save the `portlet.xml` file.

61.2.5.2 Adding a Processing Event to a Portlet

If your want a portlet to listen for a particular portlet event, define it as a processing event.

To add a processing event to a portlet:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. If the portlet event has not yet been defined within the Portlet Producer application, you must do this first, for more information, see Section 61.2.5.1, "Defining a Portlet Event at the Application Level."
3. Click the **Portlets** tab and select the portlet in which you want to use the portlet event.
4. Click the **Events** tab.

   In the Processing Events panel, the **Available** list shows all the portlet events that have been defined for the application.
5. Select the portlet event that you want to use in the portlet and click the **Add** icon to move it to the **Selected** list.

   This adds the portlet event as a processing event to the portlet definition in `portlet.xml`:

```
<portlet id="1234567890">
   ...
   <supported-processing-event>
      <name>myParam</name>
   </supported-processing-event>
   ...
</portlet>
```

6. Save the `portlet.xml` file.
7. You can now use this portlet event in the code for your portlet.

61.2.5.3 Adding a Publishing Event to a Portlet

If you want a portlet to trigger a particular portlet event, define it as a publishing event.

To add a publishing event to a portlet:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. If the portlet event has not yet been defined within the Portlet Producer application, you must do this first, for more information, see Section 61.2.5.1, "Defining a Portlet Event at the Application Level."
3. Click the **Portlets** tab and select the portlet in which you want to use the portlet event.
4. Click the **Events** tab.

   In the Publishing Events panel, the **Available** list shows all the portlet events that have been defined for the application.
5. Select the portlet event that you want to use in the portlet and click the **Add** icon to move it to the **Selected** list.

   This adds the portlet event as a publishing event to the portlet definition in `portlet.xml`:

```
<portlet id="1234567890">
   ...
   <supported-publishing-event>
      <name>myParam</name>
   </supported-publishing-parameter>
   ...
</portlet>
```

6. Save the `portlet.xml` file.
7. You can now use this portlet event in the code for your portlet.

61.2.6 How to Use Public Render Parameters

Public render parameters are a JSR 286 feature that enable portlets to share parameter values, allowing a form of inter-portlet communication.

For example, if a Map portlet and a Weather portlet are both configured to use a Zipcode public render parameter, entering a zip code in the Map portlet updates the same parameter value in the Weather portlet.

Any public render parameters supported by a portlet must be declared in the application section of the portlet deployment descriptor (`portlet.xml`). Public render parameters defined at the application level in this way are available to all the portlets in the application.

Individual portlets within the application can then specify which of these public render parameters they want to use.

This section includes the following subsections:

- Section 61.2.6.1, "Defining a Public Render Parameter at the Application Level"
- Section 61.2.6.2, "Adding a Public Render Parameter to a Portlet"

For more information about public render parameters and how to implement them, see the JSR 286 specification at:

`http://jcp.org/en/jsr/detail?id=286`

61.2.6.1 Defining a Public Render Parameter at the Application Level

For a public render parameter to be available to a portlet, it must first be declared in the application section of the portlet deployment descriptor (`portlet.xml`).

To define a public render parameter at the application level:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Parameters** tab.
3. Click the **Add** icon to create a new public render parameter.
4. From the dropdown list, select whether you are specifying a fully qualified name (QName) for the parameter, or just the local part of the name.

   A QName uniquely identifies the public render parameter across applications by specifying a namespace for the parameter as well as a local name. Within the portlet code, a public render parameter is accessed by its identifier, which identifies the parameter uniquely within the application. However, a page typically contains multiple portlets which may come from different applications. Using QNames ensures that public render parameters from one portlet do not unintentional interfere with the other portlets on a page regardless of where those portlets come from.

   If the namespace for the parameter is the same as the application default namespace, you can omit the namespace when defining the parameter by specifying an unqualified name. If the application default namespace has not been defined, a parameter with an unqualified name uses the XML default namespace.
5. If you selected **Qualified Name**:
   a. In the **Namespace** field, enter the namespace for the parameter.
   b. In the **Name** field, enter the local part of the parameter name.
6. If you selected **Unqualified Name**, in the **Name** field, enter the local part of the parameter name. The parameter uses the application default namespace as the namespace for the parameter (or the XML default namespace if no default namespace is defined for the application).
7. In the **Identifier** field, enter a name to identify the public render parameter within the application.

   The identifier must be unique within the application and is used to identify the parameters used by a portlet and in the portlet code to access the parameter. Using the identifier means that portlet developers do not need to be aware of the parameter's fully qualified name; they simply need to know the simpler application-specific identifier.
8. In the **Description** field, enter a description for the public render parameter. The description should provide enough information so that portlet developers can determine how to use this parameter in their portlets.

9. (Optional) In the Aliases panel, you can provide a list of aliases so that the parameter can accept values from other public render parameters even if they have different QNames.
   a. Click the **Create** icon to create a new alias for the parameter.
   b. In the **Namespace** field, enter the namespace of the parameter to use for the alias.
   c. In the **Name** field, enter the local part of the parameter to use for the alias.

   Note: When creating aliases for public render parameters, it is a good idea to set up reciprocal aliases. So if a parameter in portlet A has an alias to a parameter in portlet B, you should also, if possible, create an alias for the parameter in portlet B to the parameter in portlet A.

10. The public render parameter is added to the application definition section of the `portlet.xml` file:

```
<public-render-parameter>
   <description>This is my parameter</description>
   <identifier>myParam</identifier>
   <qname xmlns:x="http://example.com/params">x:myParam</qname>
   <alias xmlns:x="http://example.com/params">x:yourParam</alias>
</public-render-parameter>
```

11. Save the `portlet.xml` file.

61.2.6.2 Adding a Public Render Parameter to a Portlet

If you want a portlet to support a public render parameter, add it to the portlet.

To add a public render parameter to a portlet:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. If the public render parameter has not yet been defined within the Portlet Producer application, you must do this first, for more information, see Section 61.2.6.1, "Defining a Public Render Parameter at the Application Level."
3. Click the **Portlets** tab and select the portlet in which you want to use the public render parameter.
4. Click the **Parameters** tab.

   In the Publishing Events panel, the **Available** list shows all the portlet events that have been defined for the application.
5. Click the **Parameters** tab.

   In the Public Render Parameters panel, the **Available** list shows all the public render parameters that have been defined for the application.
6. Select the public render parameter that you want to use in the portlet and click the **Add** icon to move it to the **Selected** list.

   This adds the public render parameter to the portlet definition in `portlet.xml`:

```
<portlet id="1234567890">
 ...
 <supported-public-render-parameter>myParam</supported-public-render-parameter>
```

```
 ...
</portlet>
```

7. Save the `portlet.xml` file.
8. You can now use this public render parameter in the code for your portlet.

61.2.7 How to Add Personalization

Portlet preferences enable end users to personalize the portlet at runtime. These personalizations are visible only to the user that performed them; not to other users. By default, when you create a JSR 286 portlet using the JDeveloper wizard, a portlet preference is created to enable users to personalize the title of the portlet at runtime. You can create additional portlet preferences, either during portlet creation or by editing the `portlet.xml` file after portlet creation, to enable end users to perform other personalizations on the portlet at runtime.

This section includes the following subsections:

- Section 61.2.7.1, "Adding a Portlet Preference to a Portlet"
- Section 61.2.7.2, "Simple Portlet Personalization Example"

61.2.7.1 Adding a Portlet Preference to a Portlet

To add portlet personalization, you must create portlet preferences for those attributes of the portlet that you want users to be able to personalize.

To add a portlet preference to a portlet:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Portlets** tab and select the portlet for which you want to add personalization.
3. Click the **Advanced** tab.

   The Preferences panel lists any existing portlet preferences that exist for the portlet, for example, the `portletTitle` preference.
4. Click the **Add** icon to create a new portlet preference.
5. In the **Name** field, enter a name for the preference.

   The name must be unique within the portlet and use only letters, numbers, and the underscore character.
6. In the **Value** field, enter one or more default values for the preference. Separate multiple entries with commas.
7. Select **Read-only** if you do not want users to be able to update the value of the preference.
8. You can now use this preference in your portlet code using the `getPreference` method.

Note: If you want the preference to be translatable, you must add the appropriate key-value pairs to the resource bundle class.

61.2.7.2 Simple Portlet Personalization Example

The following example provides a simple illustration of how you can enable personalization in portlets.

To add simple personalization to a portlet:

1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Expand the project that contains the portlet.
3. Right-click `view.jsp` and choose **Open.**
4. In the visual editor, click the **Source** tab and add the code indicated in bold in Example 61–1 to display the `portletContent` customization preference:

Example 61–1 view.jsp Sample Code

```
<%@ page contentType="text/html"
    import="javax.portlet.*,java.util.*,Portlets.Portlet1,
    Portlets.resource.Portlet1Bundle"%>
<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet"%>

<portlet:defineObjects/>
<%
String[] str = {"Portlet Content"};
PortletPreferences prefs = renderRequest.getPreferences();
str = prefs.getValues("portletContent",str);
for (int i=0; i<str.length; i++)
{
%><%=(i<str.length-1)?str[i]+", ":str[i]%><%}%>
```

5. In the Application Navigator, right-click `edit.jsp` and choose **Open.**

   Notice that the JSP consists of a form field, a form input field, and two form button fields.

6. In the visual editor, click the **Source** tab and add the code indicated in bold in Example 61–2 to implement a form field for users to enter a value for the `portletContent` customization preference:

Example 61–2 edit.jsp Sample Code

```
<%@ page contentType = "text/html; charset=windows-1252"
         pageEncoding = "windows-1252"
         import = "javax.portlet.*, java.util.*,
Portletenhance.Portletenhance,
Portletenhance.resource.PortletenhanceBundle"%>
@ <%@ taglib uri = "http://java.sun.com/portlet" prefix="portlet"%>
<portlet:defineObjects/>
<%
    PortletPreferences prefs = renderRequest.getPreferences();
    ResourceBundle res =
        portletConfig.getResourceBundle(renderRequest.getLocale());
%>
<form action="<portlet:actionURL/>" method="POST">
  <table border="0">
    <tr>
      <td width="20%">
        <p class="portlet-form-field" align="right">
          <%=  res.getString(PortletenhanceBundle.PORTLETCONTENT) %>
        </p>
```

```
      </td>
      <td width="80%">
        <input class="portlet-form-input-field"
               type="TEXT"
               name="<%= Portletenhance.PORTLETCONTENT_KEY %>"
               value="<%= Portletenhance.buildValue(prefs,
Portletenhance.PORTLETCONTENT_KEY) %>"
               size="20">
      </td>
    </tr>    <tr>
      <td width="20%">
        <p class="portlet-form-field" align="right">
          <%=  res.getString(PortletenhanceBundle.PORTLETTITLE) %>
        </p>
      </td>
      <td width="80%">
        <input class="portlet-form-input-field"
               type="TEXT"
               name="<%= Portletenhance.PORTLETTITLE_KEY %>"
               value="<%= prefs.getValue(Portletenhance.PORTLETTITLE_KEY,
res.getString("javax.portlet.title")) %>"
               size="20">
      </td>
    </tr>
    <%
    String[] str = {"Portlet Content"};
    str = prefs.getValues("portletContent",str);
%>
<tr><td width="20%">
<p class="portlet-form-field" align="right">
Content
</p>
</td><td width="80%">
<textarea rows="10" cols="60" class="portlet-form-input-field"
  name="portletContent"><%
for (int i=0; i<str.length; i++)
{%><%= (i<str.length-1) ? str[i]+", " : str[i] %><%}%>
</textarea>
</td></tr>
    <tr>
      <td colspan="2" align="center">
        <input class="portlet-form-button" type="submit"

name="<%=Portletenhance.OK_ACTION%>"

value="<%=res.getString(PortletenhanceBundle.OK_LABEL)%>">
        <input class="portlet-form-button" type="submit"

name="<%=Portletenhance.APPLY_ACTION%>"

value="<%=res.getString(PortletenhanceBundle.APPLY_LABEL)%>">
      </td>
    </tr>
  </table>
</form>
```

7. In the Application Navigator, right-click *portletName*.java and choose **Open.**
8. In the visual editor, click the **Source** tab and add the following two lines of code (indicated in bold) to the `processAction` method:

```
// Save the preferences.
PortletPreferences prefs = request.getPreferences();
String param = request.getParameter(PORTLETTITLE_KEY);
prefs.setValues(PORTLETTITLE_KEY, buildValueArray(param));
String contentParam = request.getParameter("portletContent");
if (contentParam != null)
{
  prefs.setValues("portletContent", buildValueArray(contentParam));
}
prefs.store();
```

9. Redeploy the portlet.

   Notice that JDeveloper automatically saves and compiles the code before deploying the portlet. For a reminder of how to perform this step, see Chapter 62, "Testing and Deploying Your Portlets."

10. Reload the page that contains the portlet and you can see that the portlet now displays the text Portlet Content, which was one of the changes you made.

11. Click the **Personalize** link to see the form field that you added. Enter some text in this field and close the dialog. You can see the new text displayed in the portlet.

61.2.8 How to Use Portlet Filters

Portlet filters are a JSR 286 feature that enable you to alter the content of a portlet at runtime. A portlet filter is a reusable Java component that can transform the content of portlet requests and portlet responses. Filters do not generally create a response or respond to a request as portlets do, rather they modify or adapt the requests and responses.

This section includes the following subsections:

- Section 61.2.8.1, "Adding a Portlet Filter to an Application"
- Section 61.2.8.2, "Applying a Portlet Filter to a Portlet"

For more information about portlet filters and how to implement them, see the JSR 286 specification at:

`http://jcp.org/en/jsr/detail?id=286`

61.2.8.1 Adding a Portlet Filter to an Application

For a portlet to be able to use a portlet filter, the filter must first be defined within the application.

To add a portlet filter to an application:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."

2. Click the **Filters** tab.

3. Click the **Add** icon to create a new portlet filter.

4. In the Create Portlet Filter dialog, select **New Portlet Filter** to create a new filter class to implement the portlet filter. Go to step 5.

   Select **Choose from Existing Class**, and enter or browse for the filter class, if the filter class that implements the portlet filter already exists. Go to step 9.

5. In the **Name** field, enter a name for the portlet filter.

The name must be unique within the application and use only letters, numbers, and the underscore character.

6. In the **Package** field, enter or browse for the package to contain the new filter class.
7. Select one or more of the **Lifecycle Phase** checkboxes to specify which of the `javax.portlet.filter` interfaces the filter class implements:
   - **Action**—The filter class implements the `ActionFilter` interface
   - **Event**—The filter class implements the `EventFilter` interface.
   - **Render**—The filter class implements the `RenderFilter` interface.
   - **Resource**—The filter class implements the `ResourceFilter` interface.

   Note: When you create a new filter class using the Create Portlet Filter dialog, you can specify which of the filter interfaces the filter class implements. After the filter class has been created, you cannot add or remove filter interfaces through the Overview Editor. Instead, you must edit the source of the filter class directly to manually add or remove the interfaces and the `doFilter` methods defined for those interfaces.
8. Click **OK**.
9. In the **Display Name** field, enter a more user-friendly name for the portlet filter.
10. In the **Description** field, enter a description for the portlet filter.
11. In the Initialization Parameters panel, you can specify initialization parameters that pass values to the `init()` method of the filter class.
    a. Click the **Add** icon to specify an initialization parameter for the portlet filter.
    b. In the **Name** field, enter the name of the initialization parameter.
    c. In the **Value** field, enter the value to pass to the initialization parameter.
    d. In the **Description** field, enter a description of the initialization parameter.
12. The portlet filter is added to the application definition section of the `portlet.xml` file:

```
<filter>
    <display-name>Test Filter</display-name>
    <filter-name>filter_1</filter-name>
    <filter-class>javaportlets.MyFilter</filter-class>
    <lifecycle>ACTION_PHASE</lifecycle>
    <lifecycle>RENDER_PHASE</lifecycle>
</filter>
```

13. Save the `portlet.xml` file.

61.2.8.2 Applying a Portlet Filter to a Portlet

After the portlet filter has been defined in the application, you can then apply it to one or more portlets within the application. You can also specify the order in which portlet filters are applied to portlets.

To apply a portlet filter to a portlet:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."

2. Click the **Filter Mappings** tab.
3. Click the **Add** icon to add a row to the filter mappings table.
4. From the **Filter Name** dropdown list, select the portlet filter that you want to apply to the portlet.

   The dropdown list is populated with all the portlet filters that are defined in the `portlet.xml` file.
5. From the **Portlet Name** dropdown list, select the portlet to which you want to apply the portlet filter.

   The dropdown list is populated with all the portlets that are defined in the `portlet.xml` file.

   Alternatively, you can use wildcards to apply the portlet filter to more than one portlet. For example, you can enter `*` to apply the portlet filter to all the portlets in the application.
6. The filter mapping is added to the application definition section of the `portlet.xml` file:

```
<filter-mapping>
    <filter-name>filter_1</filter-name>
    <portlet-name>jsrportlet4</portlet-name>
</filter-mapping>
```

7. Use the **Top**, **Up**, **Down**, and **Bottom** icons to order the portlet filters in the `portlet.xml` file. The order of the portlet filters determines the order in which the filters are applied to the portlets.

   Note: Remember that filters can be mapped to multiple portlets. If you have multiple portlets mapped to (sharing) a filter, arbitrarily changing the filter order can produce undesired side effects. For example, if you change the order in which filters are applied in one portlet, the reordering will apply to all other portlets that share the filter.
8. Save the `portlet.xml` file.

61.2.9 How to Implement Portlet Caching

When you have completed the basic functionality of your portlet, you may want to turn your attention to portlet performance.

Caching is a common technique for enhancing the performance of web sites that include a great deal of dynamic content. The overhead involved in retrieving data and generating the output for dynamic content can be significantly reduced by proxying requests through a local agent backed by a large, low-latency data store known as a cache. The cache agent responds to a request in one of two ways, as follows:

- If a valid version of the requested content exists in the cache, then the agent simply returns the existing cache copy, thus skipping the costly process of content retrieval and generation. This condition is known as a *cache hit*.
- If a valid version of the requested content does not exist in the cache, then the agent forwards the request to its destination and awaits the return of the content. The agent returns the content to the requester and stores a local copy in its cache

for reuse if a subsequent request for the same content arises. This condition is known as a *cache miss*.

Producers generate dynamic content (that is, portlets) and they reside remotely from the Framework application instance on which they are deployed. As such, caching might improve their performance. The architecture lends itself well to caching. You can cache the portlets rendered by your producer and reuse the cached copies to handle subsequent requests, minimizing the overhead your producer imposes on page assembly.

For JSR 286 portlets, there are two different caching methods. The methods differ mainly in how they determine whether content is still valid.

- **Expiry-based caching:** When a producer receives a render request, it stamps its response with an expiry time. The rendered response remains in the cache and fills all subsequent requests for the same content until its expiry time passes. This caching scheme is perhaps the simplest and most performant because the test for cache validity requires very little overhead and does not involve any network round trips. Expiry-based caching suits applications where the content has a well-defined life span. For content with a less certain life span, however, expiry-based caching is less effective. For more information, see Section 61.2.9.1, "Implementing Expiry-Based Caching.".
- **Validation-based caching:** When a producer receives a render request, it stamps a response with a version identifier (or `ETag`). The response goes into the cache, but before the consumer can reuse the cached response, it must determine whether the cached version is still valid. It sends the producer a render request that includes the version identifier of the cached content. The producer determines whether the version identifier remains valid. If the version identifier is still valid, then the producer immediately sends a lightweight response to the consumer without any content, which indicates that the cached version can be used. Otherwise, the producer generates new content with a new version identifier, which replaces the previously cached version. In this form of caching, the consumer must always confirm with the producer whether the content is up to date. The validity of the cached copy is determined by some logic in the producer. The advantage of this approach is that the producer controls the use of the cached content, rather than relying on a fixed period. For more information, see Section 61.2.9.2, "Implementing Validation-Based Caching."

This section includes the following subsections:

- Section 61.2.9.1, "Implementing Expiry-Based Caching"
- Section 61.2.9.2, "Implementing Validation-Based Caching"

61.2.9.1 Implementing Expiry-Based Caching

You can choose to implement expiry-based caching when you first create a portlet using the Create JSR 286 Portlet Wizard. However, during the initial development of a portlet, you may prefer to turn portlet caching off and implement it later in the development cycle when the portlet content becomes more stable. You may also want to edit the expiration period, or change the cache scope.

To implement expiry-based caching:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. Click the **Portlets** tab and select the portlet for which you want to implement expiry-based caching.

3. Click the **Advanced** tab.
4. In the Cache Management panel, select **Cache Portlet**.

> **Note:** To disable portlet caching, deselect **Cache Portlet**. This sets the cache expiration period to `0`, meaning that the cached content is always expired.

5. Select the **Cache Scope**:
   - Select **Public** to share the cached content across users
   - Select **Private** if the cached content should not be shared across users.
6. Specify the cache expiration period:
   - **Select Cache Content Never Expires** to set the cache expiration period to -1. This means that the cached content never expires and the content is always retrieved from the cache. Use this option if you are confident that the portlet contains static content that is unlikely to change. Setting this option results in the following code being added to the `portlet.xml` file:

```
<portlet>
  ...
  <expiration-cache>-1</expiration-cache>
  ...
</portlet>
```

   - Select **Cache Content Expires After** to expire the cached portlet content after a specified number of seconds. Enter the expiration period, in seconds, in the adjacent field. Setting this option results in the following code being added to the `portlet.xml` file:

```
<portlet>
  ...
  <expiration-cache>300</expiration-cache>
  ...
</portlet>
```

7. Save the `portlet.xml` file.

61.2.9.2 Implementing Validation-Based Caching

Implementation of validation-based caching takes place after the initial portlet creation and requires hand coding.

Example 61–3 shows how a `GenericPortlet` would typically implement its `doView()` method such that the consumer caches the markup using validation-based caching. The example also shows how expiry-based caching can be defined programmatically (using the `CacheControl.setExpirationTime()` method) and used in conjunction with validation-based caching to further reduce the load on the producer. This would work equally well for `serveResource()`.

Example 61–3 A JSR 286 Portlet Implementing Validation-Based Caching

```
protected void doView (RenderRequest request, RenderResponse response)
  throws PortletException, IOException
{
  CacheControl cacheControl = response.getCacheControl();
  String eTag = request.getETag();
```

```
  if (isMarkupStillValid(eTag))
  {
    cacheControl.setExpirationTime(30);      // Wait 30 seconds before checking ETag again
    cacheControl.setUseCachedContent(true);  // Tell consumer to use its cached content
    return;
  }

  // ETag not valid so set a new one ...
  cacheControl.setETag(generateETag(...));   // Define a new ETag
  cacheControl.setExpirationTime(60);        // Wait 60 seconds before checking ETag again

  // ... and generate fresh portlet markup
  createMarkup(request, response);
}

private boolean isMarkupStillValid(String eTag)
{
  if (eTag == null)
  {
    return false; // No ETag was supplied
  }

  // Perform portlet specific checks for the consumer's cached
  // copy of the markup still being valid based on the given ETag
  ...
}

private String generateETag(...)
{
  // Portlet specific code to generate an ETag, for example, a hash
  // of the data on which the portlet's markup is based
  ...

  return eTag;
}
```

For more information about validation-based caching in JSR 286 portlets, see the JSR 286 specification at:

http://jcp.org/en/jsr/detail?id=286

61.2.10 How to Implement Export of Customizations (WSRP 2.0)

Another new feature that arrives with WSRP 2.0 is the ability to keep customizations with portlets when moving them from one deployment to another. Customizations are portlet preferences that are set in edit defaults mode. For example, suppose that you create a portlet and then customize its title within your development environment. If you have enabled export for that producer, then the customized title is transported along with the portlet when you deploy it in production environment. If you do not enable export, then all customizations are lost when you transport the portlet from one deployment environment to another.

To implement export of portlet customizations:

1. Open the Overview Editor for the `portlet.xml` file. For more information, see Section 61.2.1, "How to Edit the Portlet Deployment Descriptor File."
2. In the Container Runtime Options section of the Application tab, click the **Add** icon and choose **com.oracle.portlet.allowWsrpExport**.

3. In the row for the newly added container runtime option, double-click the **Value** field to edit it, and enter `true`.
4. Save the `portlet.xml` file.

61.2.11 How to Implement Rewritten URLs for Resource Proxy

Resource proxying is the standard way to retrieve resources with WSRP. To avoid problems with URLs within your portlet, you can set a flag to rewrite all of the URLs within a specified resource. For example, if have an HTML fragment that contains URLs, then you could set this flag to rewrite its URLs taking into account the WSRP resource proxying.

To indicate that URLs should be rewritten, set the `PortletRequest` attribute, `oracle.portlet.server.resourceRequiresRewriting`, to `true`. For example, you might include code similar to the excerpt in Example 61–4 to use resource proxying for a URL that you are encoding. Encapsulate this code within a method to avoid repeating it for every URL individually.

Example 61–4 Resource Proxy for WSRP

```
request.setAttribute("oracle.portlet.server.resourceRequiresRewriting",
     Boolean.TRUE);
String url = response.encodeURL(pathToResourceForRewriting);
request.removeAttribute("oracle.portlet.server.resourceRequiresRewriting");
```

If you do not specifically set `oracle.portlet.server.resourceRequiresRewriting`, then it defaults to `false`, meaning that URLs are not rewritten. You must explicitly activate the feature by setting this attribute to `true`.

61.2.12 How to Implement Stateless Resource Proxying

If you have out of protocol resources that do not require rewriting, you may want to use stateless resource proxying. Stateless resource proxying means that the URLs returned to the browser do not require portlet IDs or any other contextual information. This increases the cache hit ratio for such resources. You might find stateless resource proxying useful for functionality such as static JavaScript files, static images, and so on.

To indicate that stateless proxying is required, set the `PortletRequest` attribute `oracle.portlet.server.useStatelessProxying` to `true`. For example, you might include code similar to the excerpt in Example 61–5 to use stateless proxying for a URL that you are encoding. Encapsulate this code within a method to avoid repeating it for every URL individually.

Example 61–5 Stateless Resource Proxying

```
request.setAttribute("oracle.portlet.server.useStatelessProxying", Boolean.TRUE);
String url = response.encodeURL(pathToResource);
request.removeAttribute("oracle.portlet.server.useStatelessProxying");
```

If you do not specifically set `oracle.portlet.server.useStatelessProxying`, it defaults to `false`. You must explicitly activate the feature by setting this attribute to `true`.

61.2.13 How to Enable Java Object Cache for Preference Store Access

To improve preference store performance, you may choose to use Java Object Cache for preference store access. This avoids the need to access the persistent store on each request. You can configure the use of caching by the WSRP preference store with the following JNDI variable:

```
oracle/portal/wsrp/server/enableJavaObjectCache
```

By default, this variable is set to `false`. You can set the variable yourself in the `web.xml` file for your Portlet Producer application as follows:

```
<env-entry>
 <env-entry-name>oracle/portal/wsrp/server/enableJavaObjectCache</env-entry-name>
 <env-entry-type>java.lang.Boolean</env-entry-type>
 <env-entry-value>true</env-entry-value>
</env-entry>
```

For more information about setting JNDI variables, see Section 61.3.5.2, "Setting JNDI Variable Values."

61.2.14 How to Implement Security for JSR 286 Portlets

You can secure JSR 286 portlets that are deployed to a WSRP producer by configuring security at the WSRP producer end and the client end. For information about securing a JSR 286 portlet through its WSRP producer, see Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

61.3 Enhancing PDK-Java Portlets

When you have built your initial portlet in the Create Oracle PDK-Java Portlet wizard as described in Section 60.2.4, "How to Create a PDK-Java Portlet," the next step is to enhance it. You can find the JavaDoc reference for the PDK-Java in the *Oracle Fusion Middleware Java API Reference for Oracle PDK Java*:

This section includes the following subsections that describe some enhancements that you might want to perform:

- Section 61.3.1, "Guidelines for PDK-Java Portlets"
- Section 61.3.2, "How to Add Portlet Modes"
- Section 61.3.3, "How to Implement Public Parameters"
- Section 61.3.4, "How to Implement Private Parameters"
- Section 61.3.5, "How to Use JNDI Variables"
- Section 61.3.6, "How to Access Session Information"
- Section 61.3.7, "How to Enhance Portlet Performance with Caching"

The source code for many of the examples referenced in this section is available as part of the Portlet Developer's Kit (PDK).

When you unzip PDK-Java, find the examples in:

```
../pdk/jpdk/v2/src/oracle/portal/sample/v2/devguide
```

61.3.1 Guidelines for PDK-Java Portlets

In WebCenter Portal: Framework, PDK-Java portlets work somewhat differently than they did in Oracle Portal. As a result, you must be aware of the following new design considerations when you build PDK-Java portlets in WebCenter Portal: Framework:

- Your portlet must not contain any code that relies upon the URL format or parameters in the request that were not explicitly added by your portlet.
- You should never assume that your portlet is the only one on a page, regardless of the portlet mode. For example, even if your portlet is in Edit mode, you should not assume that it is the only portlet on the page.
- Never write a portlet mode that simply redirects. A redirect can only be issued while processing a post to your portlet or following a link generated by your portlet.

61.3.2 How to Add Portlet Modes

In the Create Oracle PDK-Java Portlet wizard, you add portlet modes by checking boxes on the wizard pages. For more information about using the wizard, see Section 60.2.4, "How to Create a PDK-Java Portlet." For each portlet mode that you select in the wizard, a basic skeleton is created. If you want to add a portlet mode after creating the portlet, you can do that manually by updating `provider.xml` and HTML or JSPs in JDeveloper.

The principles of implementing portlet modes using `RenderManager` are the same for all modes.

For more detailed information about the PDK runtime classes used in this section, see the JavaDoc in the *Oracle Fusion Middleware Java API Reference for Oracle PDK Java*:

For more information about the syntax of `provider.xml`, see the provider JavaDoc, also available on OTN.

Before You Begin

The steps that follow assume that you have:

- Built a portlet using the Create Oracle PDK-Java Portlet wizard
- Successfully registered the portlet's producer
- Added the portlet to a page

To add a portlet mode:

1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Expand the project that contains the portlet.
3. Expand the **Web Content** node and then the **htdocs** node and then the node for the producer.
4. Right-click the node for the portlet and choose **New.**

   The node for the portlet is located under **Web Content > htdocs >** ***provider***.
5. In the New Gallery, expand **Web Tier**, select **HTML** or **JSP**, and click **OK**.

   You must create an HTML file or a JSP for each mode to add to your portlet. For example, to implement Help mode, create an HTML file to provide the help content.

6. In the resulting dialog, enter a file name for the HTML file or JSP and click **OK**.
7. In the visual editor, edit the content of the page to implement the desired functionality.

   For example, for Help mode, you could add the following HTML:

   ```
   <p>This is the <i>Help</i> mode of your portlet!</p>
   ```

8. In the Application Navigator, right-click the `provider.xml` file for the provider that owns the portlet and choose **Open**.

   The `provider.xml` file is located under **Web Content > WEB-INF > providers > *provider***.
9. Change the value of the appropriate tag for the mode you are adding to `true`.

   For example, if you are adding Help mode, change the `hasHelp` tag as follows:

   ```
   <hasHelp>true</hasHelp>
   ```

   This indicates to the PDK Framework that a link or icon to that mode should be rendered.
10. Add the code to point to the HTML page or JSP that you created earlier for the mode.

    For example, for the Help page, add the following code:

    ```
    <helpPage>/htdocs/myprovider/myportlet/myHelpPage.html</helpPage>
    ```

11. Save your changes.
12. Redeploy your portlet. For more information, see Chapter 62, "Testing and Deploying Your Portlets."

    When you redeploy, JDeveloper automatically saves and compiles the code before deploying the portlet.
13. Copy the HTML file or JSP you created and the updated `provider.xml` file to the WLS instance where you plan to deploy the portlet.

    This step is not necessary if you have redeployed the producer application to a server instance.
14. Refresh the producer.
15. Refresh the page containing your portlet.

    You should now be able to access the new mode. For example, if you added Help mode, you should be able to click the Help link.

61.3.3 How to Implement Public Parameters

PDK-Java and WebCenter Portal: Framework provide public and private portlet parameters to enable portlet developers to easily write reusable, complex portlets. The Create Oracle PDK-Java Portlet wizard in JDeveloper creates portlets that are set up to use parameters. This feature enables you to focus solely on adding business logic to your portlets and does not require any changes to `provider.xml`.

For an overview of parameters, see Section 58.2.11, "Public Portlet Parameter Support" and Section 58.2.12, "Private Portlet Parameter Support."

Before You Begin

The steps that follow assume that you have:

- Followed through and understood Section 60.2.4, "How to Create a PDK-Java Portlet."
- Built a portlet using the wizard and successfully added it to a page.

> **Note:** Each portlet is limited to 4K of data. The lengths of parameter and event names, display names, and descriptions all contribute toward this 4K limit. Hence, you should not use too many parameters and events for each portlet, or give them lengthy names and descriptions.

Using the Create Oracle PDK-Java Portlet wizard, you can easily create a portlet with public parameters. When you register the producer and drop the portlet on a page, the portlet's parameters are automatically linked to page variables.

To create a portlet with public parameters:

1. In the JDeveloper Application Navigator, open the application that contains the portlet.
2. Right-click the project under which you want to create your portlet, and choose **New.**

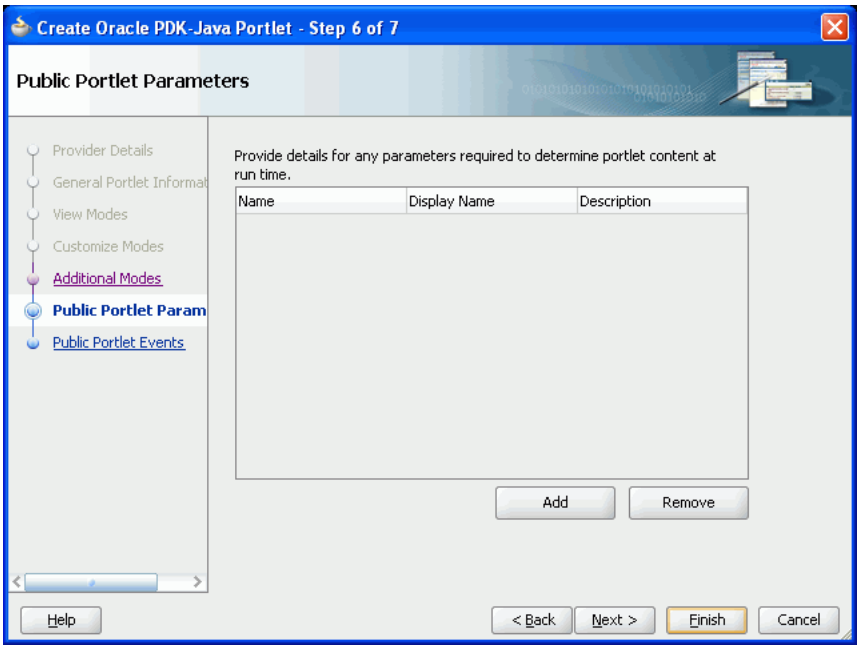
> **Note:** To create the portlet in an existing producer, right-click the producer's `provider.xml` file and choose **Add Portlet.** This takes you directly to the General Portlet Information page of the Create Oracle PDK-Java Portlet wizard.

3. In the New Gallery, expand **Web Tier,** select **Portlets** and then **Oracle PDK-Java Portlet,** and click **OK.**
4. Proceed through the Create Oracle PDK-Java Portlet wizard until you reach the Public Portlet Parameters page.

   See Section 60.2.4, "How to Create a PDK-Java Portlet" for basic information about going through the wizard.
5. On the Public Portlet Parameters page (Figure 61–2), click **Add.**

   This adds a new row to the table of parameters.

Figure 61–2 Public Portlet Parameters Page of Portlet Wizard



6. Replace the default values in the **Name, Display Name,** and **Description** fields with something more meaningful for your parameter.
7. Add more parameters as required and then click **Finish.**
8. In the Application Navigator, right-click the `provider.xml` file for the provider that owns the portlet and choose **Open.**

   The `provider.xml` file is located under **Web Content > WEB-INF > providers >** ***provider.***

   You should see entries for the parameters that you added on the Public Portlet Parameters page in the wizard, for example, as shown in Example 61–6.

Example 61–6 provider.xml Sample, Public Parameters

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<?providerDefinition version="3.1"?>
<provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
  <session>false</session>
  <passAllUrlParams>false</passAllUrlParams>
  <preferenceStore class=
     "oracle.portal.provider.v2.preference.FilePreferenceStore">
   <name>prefStore1</name>
   <useHashing>true</useHashing>
  </preferenceStore>
  <portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
      <id>1</id>
      <name>MyPortlet</name>
      <title>My Portlet</title>
      <description>My Portlet Description</description>
      <timeout>40</timeout>
      <showEditToPublic>false</showEditToPublic>
      <hasAbout>false</hasAbout>
      <showEdit>true</showEdit>
      <hasHelp>false</hasHelp>
      <showEditDefault>false</showEditDefault>
```

```
      <showDetails>false</showDetails>
      <inputParameter class=
        "oracle.portal.provider.v2.DefaultParameterDefinition">
       <name>Parameter_01</name>
       <displayName>Parameter_01</displayName>
       <description>My first parameter</description>
      </inputParameter>
      <inputParameter class=
        "oracle.portal.provider.v2.DefaultParameterDefinition">
       <name>Parameter_02</name>
       <displayName>Parameter_02</displayName>
       <description>My second parameter</description>
      </inputParameter>
      <inputParameter class=
        "oracle.portal.provider.v2.DefaultParameterDefinition">
       <name>Parameter_03</name>
       <displayName>Parameter_03</displayName>
      </inputParameter>
      <renderer class="oracle.portal.provider.v2.render.RenderManager">
         <renderContainer>true</renderContainer>
         <renderCustomize>true</renderCustomize>
         <autoRedirect>true</autoRedirect>
         <contentType>text/html</contentType>
         <showPage>/htdocs/myportlet/MyPortletShowPage.jsp</showPage>
         <editPage>/htdocs/myportlet/MyPortletEditPage.jsp</editPage>
      </renderer>
      <personalizationManager class=
        "oracle.portal.provider.v2.personalize.PrefStorePersonalizationManager">
       <dataClass>
        oracle.portal.provider.v2.personalize.NameValuePersonalizationObject
       </dataClass>
      </personalizationManager>
   </portlet>
</provider>
```

9. In the Application Navigator, right-click the *portletname*`ShowPage.jsp` for your portlet and choose **Open.**

   The file is located under **Web Content > htdocs >** ***provider*** **>** ***portlet.***

   The portlet includes logic for retrieving the parameters, for example, as shown in Figure 61–7.

Example 61–7 ShowPage.jsp Sample

```
<%@page contentType="text/html; charset=windows-1252"
        import="oracle.portal.provider.v2.render.PortletRenderRequest"
        import="oracle.portal.provider.v2.http.HttpCommonConstants"
        import="oracle.portal.provider.v2.ParameterDefinition"
%>
<%
   PortletRenderRequest pReq = (PortletRenderRequest)
      request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
%>
<P>Hello <%= pReq.getUser().getName() %>.</P>
<P>This is the <b><i>Show</i></b> render mode!</P>
<%
   ParameterDefinition[] params =
       pReq.getPortletDefinition().getInputParameters();
%>
<p>This portlet's input parameters are...</p>
```

```
<table align="left" width="50%" ><tr><td><span
class="PortletHeading1">Name</span></td><td><span
class="PortletHeading1">Value</span></td></tr>
<%
   String name = null;
   String value = null;
   String[] values = null;
   for (int i = 0; i < params.length; i++)
   {
       name = params[i].getName();
       values = pReq.getParameterValues(name);
       if (values != null)
       {
           StringBuffer temp = new StringBuffer();
           for (int j = 0; j < values.length; j++)
           {
               temp.append(values[j]);
               if (j + 1 != values.length)
               {
                   temp.append(", ");
               }
           }
           value = temp.toString();
       }
       else
       {
           value = "No values have been submitted yet.";
       }
%>
<tr>
  <td><span class="PortletText2"><%= name %></span></td>
  <td><span class="PortletText2"><%= value %></span></td>
</tr>
<%
   }
%>
</table>
```

10. Add logic to your portlet that allows it to submit parameter values entered by users.
11. Create a second portlet using the same steps that simply displays parameter values that it retrieves.
12. Register the producer with a Framework application. For more information, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."
13. Add the two portlets to a page in the application. For more information, see Section 64.3, "Adding Portlets to a Page."
14. In the Structure window of the Application Navigator, right-click an element of the page and choose **Go to Page Definition.**

    The page definition should look similar to Example 61–8. Notice the variables at the page level and the parameters at the portlet level (indicated in bold).

Example 61–8 Page Definition File Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
                version="10.1.3.38.90" id="untitled1PageDef"
                Package="view.pageDefs">
```

```
  <executables>
    <variableIterator id="variables">
      <variable Name="portlet1_Parameter_01" Type="java.lang.Object"/>
      <variable Name="portlet1_Parameter_02" Type="java.lang.Object"/>
      <variable Name="portlet1_Parameter_03" Type="java.lang.Object"/>
    </variableIterator>
    <portlet id="portlet1"
       portletInstance="/oracle/adf/portlet/PdkPortletProducer1_1153936627784
          /applicationPortlets/Portlet1_abfc5a10_010c_1000_8003_82235f50d831"
        class="oracle.adf.model.portlet.binding.PortletBinding"
        xmlns="http://xmlns.oracle.com/portlet/bindings">
      <parameters>
        <parameter name="Parameter_01" pageVariable="portlet1_Parameter_01"/>
        <parameter name="Parameter_02" pageVariable="portlet1_Parameter_02"/>
        <parameter name="Parameter_03" pageVariable="portlet1_Parameter_03"/>
      </parameters>
    </portlet>
  </executables>
</pageDefinition>
```

15. Run the application.
16. Enter values in the first portlet and the same values should be displayed in the second portlet.

61.3.4 How to Implement Private Parameters

In some cases, you might need a parameter that is known only to the portlet instance. These parameters are known as private parameters because they have no connection to the page and are known only to the portlet. Private parameters often come in handy when you are building navigation for your portlet. For example, if you have a portlet made up of multiple pages, then you can use these private parameters to jump to another resource of the portlet.

This section includes the following subsections:

- Section 61.3.4.1, "About Private Parameters"
- Section 61.3.4.2, "About Portlet URL Types"
- Section 61.3.4.3, "Building Links with the Portlet URL Types"
- Section 61.3.4.4, "Building Forms with the Portlet URL Types"
- Section 61.3.4.5, "Implementing Navigation within a Portlet"
- Section 61.3.4.6, "Restricting Navigation to Resources"

61.3.4.1 About Private Parameters

Private parameters are used in classic web applications to pass information from links or forms in the browser back to the server. The server in turn takes actions and returns the appropriate content. For example, if the user of a dictionary web site asks for information about hedgehogs, then the URL submitted to the server might append a private parameter as follows:

```
http://dictionary.reference.com/search?q=Hedgehog
```

If the server is responsible for rendering the whole page and the client communicates directly with the server, then this form of URL works well. In a Framework application, the client does not communicate directly with portlets. Instead, WebCenter Portal: Framework mediates between the client and the portlet. Moreover,

because most pages have multiple portlets, WebCenter Portal: Framework communicates with multiple portlets.

For example, suppose a page contains two portlets, a thesaurus portlet and a dictionary portlet. Both portlets use `q` as a parameter to record the search queries made by the user. If the user queries the thesaurus portlet, then the URL used to rerequest the page with the updated thesaurus portlet must contain the thesaurus portlet's parameter, `q`. The thesaurus parameter must also be distinguished from dictionary portlet parameter 1, which performs the same function for that portlet.

You must ensure that the portlet meets the following criteria:

- It properly qualifies its own parameters when they are built into links and forms.
- It leaves unchanged any parameters that do not belong to it.

The following API call transforms an unqualified parameter name into a qualified parameter name:

```
HttpPortletRendererUtil.portletParameter(HttpServletRequest request, String
param);
```

`HttpPortletRendererUtil` is in the package `oracle.portal.provider.v2.render.http`.

For example:

```
qualParamQ = HttpPortletRendererUtil.portletParameter(r, "q");
```

To fetch the value of a portlet parameter from the incoming request, you can use the following API:

> **Note:** The API converts the parameter name into the qualified parameter name before fetching the value from the incoming request. Hence, you need not perform this step.

```
PortletRenderRequest.getQualifiedParameter(String name)
```

`PortletRenderRequest` is in the package `oracle.portal.provider.v2.render`.

For example:

```
valueQ = r.getQualifiedParameter("q");
```

The other aspect of a portlet's responsibilities with private parameters is to not disturb the parameters on the URL that it does not own. The utilities you may use to ensure adherence to this rule are discussed in Section 61.3.4.3, "Building Links with the Portlet URL Types" and Section 61.3.4.4, "Building Forms with the Portlet URL Types."

61.3.4.2 About Portlet URL Types

When a portlet renders itself, WebCenter Portal: Framework passes it various URLs, which the portlet can then use to render links. You can fetch and manipulate these URLs to simplify the task of creating links. The following is a list of the URLs provided to portlets:

- **PAGE_LINK** is a URL to the page upon which the portlet instance resides. You use this URL as the basis for all intraportlet links. If the portlet renders a link that navigates the user to another section of the same portlet, then this navigation must be encoded as a set of parameters using the PAGE_LINK.

- **DESIGN_LINK** is a URL to the portlet's personalization (Edit mode) page. A portlet's Edit and Edit Defaults modes are not rendered on the same page as the portlet. The Edit and Edit Defaults modes take over the entire browser window. The portlet's Edit and Edit Defaults modes are not necessarily accessible to every user. It represents a minimal, static framework in which the portlet is free to render its personalization options. This URL is only of use when rendering Personalize links.
- **BACK_LINK** is a URL to a useful return point from the current page where the portlet renders itself. For example, when the portlet is rendering its personalization page (Edit mode), this link refers to the page on which the portlet resides and from which the user navigated to the personalization page. Consequently, it is the link you encode in the buttons that accept or cancel the pending action. This URL is only useful for the desktop rendering of portlets (usually in Edit or Edit Defaults mode).

61.3.4.3 Building Links with the Portlet URL Types

To build links with Portlet URL types, you must access them and use them when writing portlet rendering code. To fetch the URL for a link, you call the following APIs in PDK-Java:

```
portletRenderRequest.getRenderContext().getPageURL()
portletRenderRequest.getRenderContext().getEventURL()
portletRenderRequest.getRenderContext().getDesignURL()
portletRenderRequest.getRenderContext().getLoginServerURL()
portletRenderRequest.getRenderContext().getBackURL()
```

In portlet navigation, you must add (or update) your portlet's parameters in the page URL. To perform this task, you can use the following API to build a suitable URL:

```
UrlUtils.constructLink(
    PortletRenderRequest pr,
    int linkType, -- UrlUtils.PAGE_LINK in this case
    NameValue[] params,
    boolean encodeParams,
    boolean replaceParams)
```

`UrlUtils` resides in the package called `oracle.portal.provider.v2.url`. Notice that you do not actually fetch the page URL yourself. Rather you use the supplied portlet URL type, `UrlUtils.PAGE_LINK`.

The parameter names in the `params` argument should be fully qualified. Moreover, if you properly qualify the parameters, `UrlUtils.constructLink` with the appropriate `linkType` does not disturb other URL parameters that are not owned by the portlet.

An alternative version of `UrlUtils.contructLink` accepts a URL as the basis for the returned URL. If you require an HTML link, then you can use `UrlUtils.constructHTMLLink` to produce a complete anchor element.

The following example portlet, `ThesaurusLink.jsp`, uses the parameter `q` to identify the word for which to search the thesaurus. It then creates links on the found, related words that the user may follow to get the thesaurus to operate on that new word. To see the initial submission form that sets the value of `q`, see the example in Section 61.3.4.4, "Building Forms with the Portlet URL Types."

```
<%
    String paramNameQ = "q";
    String qualParamNameQ =
    HttpPortletRendererUtil.portletParameter(paramNameQ);
    PortletRenderRequest pRequest = (PortletRenderRequest)
        request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    String paramValueQ = pRequest.getQualifiedParameter(paramNameQ);
%>
<!-- Output the HTML content -->
<center>
    Words similar to <%= paramValueQ %>
    <br>
    Click the link to search for words related to that word.
    <br>
    <ul>
<%
        String[] relatedWords = Thesaurus.getRelatedWords(paramValueQ);
        NameValue[] linkParams = new NameValue[1];
        for (int i=0; i<=relatedWords.length; i++)
        {
            linkParams[0] = new NameValue(
                qualParamNameQ, relatedWords[i]);
%>
            <li>
            <b> <%= relatedWords[i] %> </b>
            <%= UrlUtils.constructHTMLLink(
                pRequest,
                UrlUtils.PAGE_LINK,
                "(words related to " + relatedWords[i] + ")",
                "",
                linkParams,
                true,
                true)%>
           </li>
<%
        }
%>
   </ul>
</center>
```

61.3.4.4 Building Forms with the Portlet URL Types

The use of portlet parameters in forms is similar to their use in links. The following two fundamental rules continue to apply:

- Qualify the portlet's parameter names.
- Do not manipulate or remove the other parameters on the incoming URL.

In terms of markup and behavior, forms and links differ quite considerably. However, just as with links, PDK-Java contains utilities for complying with these two basic rules.

A parameter name is just a string, whether it is a link on a page or the name of a form element. Therefore, the code for properly qualifying the portlet's parameter name is the same as that described in Section 61.3.4.3, "Building Links with the Portlet URL Types."

Forms differ from links in the way you ensure that the other parameters in the URL remain untouched. Once you open the form in the markup, you can make use of the following APIs:

```
UrlUtils.htmlFormHiddenFields(pRequest,UrlUtils.PAGE_LINK, formName);
UrlUtils.htmlFormHiddenFields(someURL);
```

where `formName = UrlUtils.htmlFormName(pRequest,null).`

> **Note:** Just as parameters in URLs and element names in forms require qualification to avoid clashing with other portlets on the page, form names must be fully qualified because any given page might have several forms on it.

The `htmlFormHiddenFields` utility writes HTML hidden form elements into the form, one form element for each parameter on the specified URL that is not owned by the portlet.

```
<INPUT TYPE="hidden" name="paramName" value="paramValue">
```

Thus, you need only to add their portlet's parameters to the form.

The other item of which you should be aware is how to derive the submission target of your form. In most cases, the submission target is the current page:

```
formTarget = UrlUtils.htmlFormActionLink(pRequest,UrlUtils.PAGE_LINK)
```

The value of `formTarget` can be the action attribute in an HTML form or the target attribute in a `SimpleForm`. Even though the method name includes HTML, it actually just returns a URL and thus you can use it in mobile portlets, too.

The following example form renders the thesaurus portlet's submission form. For the portlet that results from the submission of this form, see the example in Section 61.3.4.3, "Building Links with the Portlet URL Types."

```
<%
    String paramNameSubmit = "submit";
    String paramNameQ = "q";
    String qualParamNameQ =
        HttpPortletRendererUtil.portletParameter(paramNameQ);
    String qualParamNameSubmit =
    HttpPortletRendererUtil.portletParameter(paramNameSubmit);
    PortletRenderRequest pRequest = (PortletRenderRequest)
        request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    String formName = UrlUtils.htmlFormName(pRequest,"query_form");
%>
<!-- Output the HTML content -->
<center>
    <b>Thesaurus</b>
    Enter the word you want to search for
    <form name="<%= formName %>" method="POST"
        action="<%= UrlUtils.htmlFormActionLink(pRequest,UrlUtils.PAGE_LINK) %>">
        <%= UrlUtils.htmlFormHiddenFields(pRequest,UrlUtils.PAGE_LINK, formName)%>
        <table><tr><td>
            Word of interest:
        </td><td>
            <input
                type="text"
                size="20"
                name="<%= qualParamNameQ %>"
                value="">
        </td></tr></table>
        <input type=submit name="<%= qualParamNameSubmit %>" Value="Search">
```

```
    </form>
</center>
```

61.3.4.5 Implementing Navigation within a Portlet

You can implement navigation within a portlet in one of three ways:

- Pass navigation information in rendered URLs using private portlet parameters. Branching logic within the portlet code then determines which section of the portlet to render based on the URL. This option represents a small extension to the thesaurus example presented in Section 61.3.4.3, "Building Links with the Portlet URL Types" and Section 61.3.4.4, "Building Forms with the Portlet URL Types." Basically, instead of performing thesaurus search operations using the value of parameter `q`, the portlet branches based on the parameter value and renders different content accordingly.
- Pass navigation information as described in the previous item but use PDK-Java to interpret the parameter and thus branch on its value. This option requires some further changes to the thesaurus example and is more fully explained later in this section.
- Use session storage to record the portlet state and private parameters to represent actions rather than explicit navigation. This method provides the only way that you can restore the portlet to its previous state when the user navigates off the page containing the portlet. Once the user leaves the page, all private portlet parameters are lost and you can only restore the state from session storage, assuming you previously stored it there. This option requires that you understand and implement session storage. For more information about implementing session storage, see Section 61.3.6, "How to Access Session Information."

The following portlet code comes from the multipage example in the sample producer of PDK-Java:

```
<portlet>
    <id>11</id>
    <name>Multipage</name>
    <title>MultiPage Sample</title>
    <shortTitle>MultiPage</shortTitle>
    <description>
        This portlet depicts switching between two screens all
        in an application page.
    </description>
    <timeout>40</timeout>
    <timeoutMessage>MultiPage Sample timed out</timeoutMessage>
    <renderer class="oracle.portal.provider.v2.render.RenderManager">
        <contentType>text/html</contentType>
        <showPage>/htdocs/multipage/first.jsp</showPage>
        <pageParameterName>next_page</pageParameterName>
    </renderer>
</portlet>
```

> **Note:** The value of `pageParameterName` is the name of a portlet parameter, `next_page`, that the PDK-Java framework intercepts and interprets as an override to the value of the `showPage` parameter. If the PDK-Java framework encounters the qualified version of the parameter when the multipage portlet is requested, then it renders the resource identified by `next_page` rather than `first.jsp`. PDK-Java does not render the parameter within the portlet, that responsibility falls to the portlet.

You can modify the thesaurus example to operate with the use of this parameter. Specifically, you can use the form submission portlet to be the input for the thesaurus (the first page of the portlet), then navigate the user to the results page, which contains links to drill further into the thesaurus. The following examples illustrate these changes.

> **Note:** The example that follows is most useful for relatively simple cases, such as this thesaurus example. If your requirements are more complex (for example, you want to build a wizard experience), then you should consider using an MVC framework such as Struts. For information about how to build portlets from struts applications, see Section 61.5, "Creating a Struts Portlet."

`ThesaurusForm.jsp`:

```
<%
    PortletRenderRequest pRequest = (PortletRenderRequest)
        request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    String paramNameSubmit = "submit";
    String paramNameQ = "q";
    String qualParamNameQ =
        HttpPortletRendererUtil.portletParameter(pRequest, paramNameQ);
    String qualParamNameSubmit =
    HttpPortletRendererUtil.portletParameter(pRequest, paramNameSubmit);
    String formName = UrlUtils.htmlFormName(pRequest,"query_form");
%>
<!-- Output the HTML content -->
<center>
    <b>Thesaurus</b>
    Enter the word you want to search for
    <form name="<%= formName %>" method="POST"
        action="<%= UrlUtils.htmlFormActionLink(pRequest,UrlUtils.PAGE_LINK) %>">
        <%= UrlUtils.htmlFormHiddenFields(pRequest,UrlUtils.PAGE_LINK, formName)
%>
        <%= UrlUtils.emitHiddenField(
                HttpPortletRendererUtil.portletParameter(request, "next_page"),
                "htdocs/path/ThesaurusLink.jsp" ) %>
        <table><tr><td>
            Word of interest:
        </td><td>
            <input
                type="text"
                size="20"
                name="<%= qualParamNameQ %>"
                value="">
        </td></tr></table>
        <input type=submit name="<%= qualParamNameSubmit %>" Value="Search">
```

```
    </form>
</center>
```

Notice how `next_page` must be explicitly set to point to `ThesaurusLink.jsp`. If you do not explicitly set `next_page` in this way, then it defaults to the resource registered in `provider.xml`, which is `ThesaurusForm.jsp`.

`ThesaurusLink.jsp`:

```
<%
    PortletRenderRequest pRequest = (PortletRenderRequest)
        request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    String paramNameQ = "q";
    String paramNameNextPage = "next_page";
    String qualParamNameQ =
    HttpPortletRendererUtil.portletParameter(pRequest, paramNameQ);
    String qualParamNameNextPage =
        HttpPortletRendererUtil.portletParameter(pRequest, paramNameNextPage);
    String paramValueQ = pRequest.getQualifiedParameter(paramNameQ);
%>
<!-- Output the HTML content -->
<center>
    Words similar to <%= paramValueQ %>
    <br>
    Click the link to search for words related to that word.
    <br>
    <ul>
<%
        Thesaurus t = new Thesaurus();
        String[] relatedWords = t.getRelatedWords(paramValueQ);
        NameValue[] linkParams = new NameValue[2];
        linkParams[0] = new NameValue(
            qualParamNameNextPage, "htdocs/path/ThesaurusLink.jsp");
        for (int i=0; i<relatedWords.length; i++)
        {
            linkParams[1] = new NameValue(
                qualParamNameQ, relatedWords[i]);
%>
            <li>
            <b> <%= relatedWords[i] %> </b>
            <%= UrlUtils.constructHTMLLink(
                pRequest,
                UrlUtils.PAGE_LINK,
                "(words related to " + relatedWords[i] + ")",
                "",
                linkParams,
                true,
                true)%>
           </li>
<%
        }
%>
   </ul>
   <a href="<%=XMLUtil.escapeXMLAttribute
               (pRequest.getRenderContext().getPageURL())%>">
      Reset Portlet
   </a>
</center>
```

61.3.4.6 Restricting Navigation to Resources

One limitation of implementing navigation with private parameters is that users could potentially navigate to portlet resources that you prefer to restrict. To control navigation to restricted resources, you can create a *whitelist* of acceptable resources to which a user may navigate. If you do not construct a whitelist to restrict navigation, then your portlet's resources are accessible according to the following default rules:

- Any path immediately beneath the servlet root context is navigable. For example, `/index.jsp` is accessible but `/WEB-INF/web.xml` is not.
- Any path under the `htdocs` directory is navigable. For example, both `/htdocs/multipage/first.jsp` and `/htdocs/lottery/lotto.jsp` are accessible.

To change this default behavior, you can add allowable path values to the provider definition file, `provider.xml`. For example, suppose you have a portlet where a JSP is used as a controller to forward requests to other pages depending on the `pageParameterName` private parameter. The XML excerpt in Example 61–9 allows resources under `/htdocs/multiportlet` to be shown. All other resources are restricted.

Example 61–9 Whitelist Excerpt from the provider.xml File

```
<portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
   <id>1</id>
   <name>Multipage</name>
   <title>A MultiPage Portlet</title>
   ...
   <renderer class="oracle.portal.provider.v2.render.RenderManager">
          <contentType>text/html</contentType>
          <showPage>/htdocs/multiportlet/controller.jsp</showPage>
          <pageParameterName>show_page</pageParameterName>
          <allowedPath>/htdocs/multiportlet/*</allowedPath>
   </renderer>
</portlet>
```

The pattern matching rules for this feature are similar to URL pattern matching in `web.xml` files. The rules are as follows:

- To match the defined patterns, the resource path must exactly match unless wildcards are used.
- The first wildcard is for path matching and consists of a string beginning with `/` and ending with `/*`. Any resource whose path starts with this string is matched. For an `<allowedPath>` value of `/htdocs/sub1/*`, valid values of the private parameter include `/htdocs/sub1/file.jsp` and `/htdocs/sub1/sub2/file2.jsp`.
- The second wildcard is for file type matching and consists of a string starting with `*.` and ending with a file extension. Valid values for the page parameter end with that file extension. For an `<allowedPathvalue>` of `*.jsp`, valid values of the private parameter include `/htdocs/sub1/file.jsp` and `/htdocs/sub1/file2.jsp`.

61.3.5 How to Use JNDI Variables

When writing Java portlets, you may set deployment-specific properties through the JNDI service such that their values may be retrieved from your producer code. In this

way, you can specify any property in a producer deployment and then easily access it anywhere in your producer code.

You can use JNDI variables to change producer property values after the producer has been deployed. The environment entry must be declared in `web.xml`. It can then be updated on deployment using a deployment plan.

PDK-Java provides utilities to enable the retrieval of both producer and non-producer JNDI variables within a Java EE container.

This section includes the following subsections:

- Section 61.3.5.1, "Declaring JNDI Variables"
- Section 61.3.5.2, "Setting JNDI Variable Values"
- Section 61.3.5.3, "Retrieving JNDI Variables"

61.3.5.1 Declaring JNDI Variables

You declare JNDI variables in the `web.xml` file for your producer. The format for declaring a JNDI variable is as follows:

```
<env-entry>
    <env-entry-name>variableName</env-entry-name>
    <env-entry-type>variableType</env-entry-type>
    <env-entry-value>variableValue</env-entry-value>
</env-entry>
```

The `env-entry-name` element contains the name by which you want identify the variable. `env-entry-type` contains the fully qualified Java type of the variable. `env-entry-value` contains the variable's default value.

This section includes the following subsections:

- Section 61.3.5.1.1, "Variable Types"
- Section 61.3.5.1.2, "Variable Naming Conventions"
- Section 61.3.5.1.3, "Examples"

61.3.5.1.1 Variable Types In the `env-entry-type` element, you must supply the fully qualified Java type of the variable, which is expected by your Java code. The Java types you may use in your JNDI variables are as follows:

- `java.lang.Boolean`
- `java.lang.String`
- `java.lang.Integer`
- `java.lang.Double`
- `java.lang.Float`

The Java EE container uses these type declarations to automatically construct an object of the specified type and gives it the specified value when you retrieve that variable in your code.

61.3.5.1.2 Variable Naming Conventions The PDK-Java defines environment variables that can be set at the individual producer service level or at the web application level. To avoid naming conflicts between different producer services or different application components packaged in the same web application, Oracle recommends you devise some naming convention.

> **Note:** If you use the `EnvLookup` method, then you must use `oracle/portal/provider/service/property`. You cannot substitute your own company name or component in this case.

For example:

- Producer service-specific names should be of the form:

  company`/`*component name*`/`*producer name*`/`*variable name*

- Shared names must be of the form:

  company`/`*component name*`/`*producer name*`/global`

where:

- *company* is the name of the company owning the application.
- *component name* is the name of the application or component with which the producer is associated.
- *producer name* is the service name of the producer.
- *variable name* is the name of the variable itself.

As you can see, these naming conventions are similar to those used for Java packages. This approach minimizes the chance of name collisions between applications or application components. PDK-Java provides utilities that enable you to retrieve variables in this form without hard coding the service name of the producer into your servlets or JSPs. The service name need only be defined in the producer's WAR file. For more information about retrieving JNDI variables, see Section 61.3.5.3, "Retrieving JNDI Variables."

61.3.5.1.3 Examples The following examples illustrate producer variable names:

```
oracle/portal/myProvider/myDeploymentProperty
oracle/portal/myprovider/myProperties/myProperty
```

The following example illustrates non-producer variable names:

```
oracle/portal/myOtherProperty
```

61.3.5.2 Setting JNDI Variable Values

In your producer deployment, you may want to set a new value for some or all of your JNDI variables. You can perform this task by setting the values manually within a WLS deployment plan. Deployment plans can be created for the producer deployment through the WLS console.

To set variable values manually within a deployment plan:

1. Go to the producer deployment with the WLS console and create a new deployment plan if one does not exist.
2. Edit the deployment plan XML file. For each deployment property you want to set, add the following variable definition directly under the `<deployment-plan>` tag:

```
<variable-definition>
  <variable>
    <name>jndi_var_def</name>
    <value>false</value>
  </variable>
```

```
</variable-definition>
```

3. To tie this variable definition to the actual JNDI variable, add the following for each property under the `WEB-INF/web.xml` module descriptor (`oracle/portal/sample/rootDirectory` is used as an example):

```
<module-descriptor external="false">
  <root-element>web-app</root-element>
  <uri>WEB-INF/web.xml</uri>
  <variable-assignment>
    <name>jndi_var_def</name>

<xpath>/web-app/env-entry/[env-entry-name="oracle/portal/sample/rootDirectory"]
/env-entry-value</xpath>
   </variable-assignment>
</module-descriptor>
```

4. Save and close the file.
5. Select Update on the producer deployment to apply the deployment plan for the new settings to take effect.

61.3.5.3 Retrieving JNDI Variables

JNDI is a standard Java EE technology. As such, you can access JNDI variables through Java EE APIs. For example:

```
String myVarName = "oracle/portal/myProvider/myVar"
String myVar = null;
try
{
   InitialContext ic = new InitialContext();
   myVar = (String)ic.lookup("java:env/" + myVarName);
}
catch(NamingException ne)
{
   exception handling logic
}
```

In addition to the basic Java EE APIs, PDK-Java includes a simple utility class for retrieving the values of variables defined and used by the PDK itself. These variables conform to the naming convention described in Section 61.3.5.1.2, "Variable Naming Conventions" and are of the form:

```
oracle/portal/provider_service_name/variable_name
oracle/portal/variable_name
```

To use these APIs, you need only provide the *`provider_service_name`* and the *`variable_name`*. The utilities construct the full JNDI variable name, based on the information you provide, and look up the variable using code similar to that shown earlier and return the value of the variable.

The `EnvLookup` class (`oracle.portal.utils.EnvLookup`) provides two `lookup()` methods. One retrieves producer variables and the other retrieves non-producer variables. Both methods return a `java.lang.Object`, which can be cast to the Java type you are expecting.

The following code example illustrates the retrieval of a producer variable:

```
EnvLookup el = new EnvLookup();
String s = (String)el.lookup(myProviderName, myVariableName);
```

myProviderName represents the service name for your producer, which makes up part of the variable name. *myVariableName* represents the portion of the variable name that comes after the producer's service name. The example assumes the variable being retrieved is of type `java.lang.String`.

To retrieve a non-producer variable, you use the same code, you pass only one parameter, the variable name, to the `lookup()`, again excluding the `oracle/portal` prefix.

```
EnvLookup el = new EnvLookup();
Object o = el.lookup(myVariableName);
```

Table 61–2 shows the JNDI variables provided by default with PDK-Java. If you do not declare these variables, then PDK-Java looks for their values in their original locations (`web.xml` and the deployment properties file).

Table 61–2 PDK-Java JNDI Variables

Variable	Description
`oracle/portal/provider/`*provider_name*`/autoReload`	Boolean auto reload flag. Defaults to true.
`oracle/portal/provider/`*provider_name*`/definition`	Location of producer's definition file.
`oracle/portal/provider/global/log/logLevel`	Log setting (0 through 8). 0 being no logging and 8 the most possible logging.
`oracle/portal/provider/`*provider_name*`/maxTimeDifference`	Producer's HMAC time difference.
`oracle/portal/provider/<service_name>/resourceUrlKey`	Authentication key for resource proxying through the Parallel Page Engine. See *Oracle Fusion Middleware Administrator's Guide for Oracle Portal* for more information.
`oracle/portal/provider/`*provider_name*`/rootDirectory`	Location for producer personalizations. No default value.
`oracle/portal/provider/`*provider_name*`/sharedKey`	HMAC shared key. No default value.
`oracle/portal/provider/`*provider_name*`/showTestPage`	(non-producer) A Boolean flag that determines if a producer's test page is accessible. Defaults to true.
`oracle/portal/provider/global/transportEnabled`	A Boolean flag that determines whether Edit Defaults personalizations may be exported and imported.

61.3.6 How to Access Session Information

When a user accesses a page, it initiates a public, unauthenticated session and tracks information about the session across requests. If the user logs in, then this session becomes an authenticated session of the logged-in user. This session terminates when any of the following occur:

- The browser session terminates (that is, the user closes all the browser windows).
- The user explicitly logs out.
- The session times out because the user's idle time exceeds the configured limit.

As part of the metadata generation, all of the producers that contribute portlets to the page are contacted, if they specified during registration that they be called for some special processing. This call allows producers to do processing based on the user session, log the user in the producer's application if needed, and establish producer sessions. For producers, this call is referred to as `initSession`. As most web-enabled applications track sessions using cookies, this API call enables the producer of the application to return cookies.

You can use the session store to save and retrieve information that persists during the portal session. This information is only available, and useful, to you during the life of the session. You should store only temporary information in the session store. Application developers may use the session store to save information related to the current user session. Data in the session store can be shared across portlets.

If the information you want to store must persist across sessions, then you may want to store it in the preference store instead. Some common applications of the session store are as follows:

- To cache data that is expensive to load or calculate (for example, search results).
- To cache the current state of a portlet (for example, the current range, or page, of search results displayed in the portlet, or sequence of events performed by user).

Before you implement session storage, you should carefully consider the performance costs. Because portlets and producers are remote, it can be a relatively expensive operation to create and maintain even a small amount of information in the session store. For this reason, you may want to avoid altogether any session storage for public pages that are accessed frequently by many users.

Furthermore, while using the session store with producers, you create a stateful application that tracks state information in memory. Similarly, you create a stateful application if you use the file-system implementation of preference store.

If scalability is an important concern for you, then a stateful application may cause you problems. Stateful applications can affect the load-balancing and failover mechanism for your configuration. Even though you may deploy multiple middle-tiers, you must implement sticky routing (where the same node handles subsequent requests in the same session) to track state. Sticky routing may result in lopsided load-balancing or loss of session data in case a node crashes, affecting failover. This issue is one reason why many developers prefer to build stateless applications. However, if scalability is not a concern, then a stateful application should present no problems for you.

The PDK Framework represents the session with a `ProviderSession` object, which is established during the call to the Provider Instance's `initSession` method. This object is associated with the `ProviderUser`. To make data persistent between requests, you must write data into the session object using the `setAttribute` method on the `ProviderSession` object. This method maps a `java.lang.Object` to a `java.lang.String` and stores that mapping inside the session object. The `String` can then be used to retrieve the `Object` during a subsequent request, provided the session is still valid.

A producer session may become invalid for the following reasons:

- The session times out.
- The `invalidate` method on `ProviderSession` is called.

- The JVM process running the servlet container is terminated.

All portlets contained by the same `ProviderInstance` share the same session for a particular `ProviderUser`. Therefore, data unique to a particular portlet instance must be mapped to a unique `String` in the session. This is accomplished using the `portletParameter` method in the `PortletRendererUtil` class. This method makes a supplied `String` parameter or attribute name unique to a `PortletInstance`, by prefixing it with a generated identifier for that instance. You can use the returned instance-specific name to write portlet instance data into the session.

For more detailed information about the PDK Framework classes, see the JavaDoc in the *Oracle Fusion Middleware Java API Reference for Oracle PDK Java*.

In the example in this section, session storage is used to count the number of times your portlet has rendered in Shared Screen mode.

Before You Begin

The steps that follow assume that you have:

- Followed through and understood Section 60.2.4, "How to Create a PDK-Java Portlet."
- Built a portlet using the wizard and successfully added it to a page.

To implement session storage:

- Import `ProviderSession`, `PortletRendererUtil`, and `HttpPortletRendererUtil`.
- Retrieve the producer session.
- Read and write the session by accessing it from within your Java portlet.
- Set the session to true in `provider.xml`.
- Register the producer for session storage and set the Login Frequency.

The steps that follow describe how to add a session count to your portlet that displays how many times the portlet has been rendered for the current session.

1. After using the wizard to create a portlet, you can edit the JSP for the Show page in Oracle JDeveloper. You must import the following classes:

```
<%@page contentType="text/html; charset=windows-1252"
import="oracle.portal.provider.v2.render.PortletRenderRequest"
import="oracle.portal.provider.v2.http.HttpCommonConstants"
import="oracle.portal.provider.v2.ProviderSession"
import="oracle.portal.provider.v2.render.PortletRendererUtil"
import="oracle.portal.provider.v2.render.http.HttpPortletRendererUtil"
%>
```

2. Insert code that checks for a valid session first and then increments the count and displays it. If the session is valid and a previously stored value exists, then you display the value, increment the count, and store the new value. If the session is valid but no previously stored value exists, then you initialize a new count starting with 1, and display and store the value. You also want to obtain the unique string key for this portlet and then use an it in an array to count the session. If no session information was received, then you want to provide information to the user indicating they may need to log in again.

```
<%
PortletRenderRequest pReq = (PortletRenderRequest)
```

```
request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
ProviderSession pSession = pReq.getSession();
  if (pSession != null)
  {
    String key = PortletRendererUtil.portletParameter(pReq, "count");
    Integer i = (Integer)pSession.getAttribute(key);
    if (i == null)
    {
      i = new Integer(0);
    }
    i = new Integer(i.intValue()+1);
    pSession.setAttribute(key, i);
%>

<p>Render count in this session: <%=i%> </p>

<%
  }
  else
  {
%>

<p>The session has become invalid</p>
<br>
Please log out and log in again.
<%
  }
%>
```

3. By default, the wizard does not set session to true in `provider.xml`. You must update this flag in order for the producer to receive session information from the portal. You should only set this tag to true if you are using session information in your producer or portlets. By setting this flag to true, extra load is added to the producer calls.

   ```
   <provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
   <session>true</session>
   ```

4. Register your producer with session support. For a reminder on how to register your portlet, see Section 62.4, "Registering and Viewing Your Portlet."

5. If you have not added your Java portlet to a page, then do so now. Ensure that you perform the following tasks:

   - Refresh the producer to accept the new changes.
   - Re-login in case your session is no longer valid.

61.3.7 How to Enhance Portlet Performance with Caching

Once you have completed the basic functionality of your portlet, you may want to turn your attention to portlet performance.

Caching is a common technique for enhancing the performance of web sites that include a great deal of dynamic content. The overhead involved in retrieving data and generating the output for dynamic content can be significantly reduced by proxying requests through a local agent backed by a large, low-latency data store known as a cache. The cache agent responds to a request in one of two ways, as follows:

- If a valid version of the requested content exists in the cache, then the agent simply returns the existing cached copy, thus skipping the costly process of content retrieval and generation. This condition is called a cache hit.
- If a valid version of the requested content does not exist in the cache, then the agent forwards the request to its destination and awaits the return of the content. The agent returns the content to the requester and stores a local copy in its cache for reuse if a subsequent request for the same content arises. This condition is called a cache miss.

Producers generate dynamic content (that is, portlets) and they reside remotely from the Framework application instance on which they are deployed. As such, caching might improve their performance. The architecture lends itself well to caching. You can cache the portlets rendered by your producer and reuse the cached copies to handle subsequent requests, minimizing the overhead your producer imposes on page assembly.

The producer can use any one of three different caching methods, depending upon which one is best suited to the application. The methods differ chiefly in how they determine whether content is still valid. Following are the three caching methods:

- **Expiry-based Caching**: When a producer receives a render request, it stamps its response with an expiry time. The rendered response remains in the cache and fills all subsequent requests for the same content until its expiry time passes. This caching scheme is perhaps the simplest and most performant because the test for cache validity requires very little overhead and does not involve any network round trips. Expiry-based caching suits applications where the content has a well-defined life span. For content with a less certain life span, however, expiry-based caching is less effective. For more information, see Section 61.3.7.1, "Activating Caching" and Section 61.3.7.2, "Adding Expiry-Based Caching."
- **Validation-based Caching**: When a producer receives a render request, it stamps its response with a version identifier (or E Tag). The response goes into the cache, but, before the consumer can reuse the cached response, it must determine whether the cached version is still valid. It sends the producer a render request that includes the version identifier of the cached content. The producer determines whether the version identifier remains valid. If the version identifier is still valid, then the producer immediately sends a lightweight response to the consumer without any content, which indicates the cached version can be used. Otherwise, the producer generates new content with a new version identifier, which replaces the previously cached version. In this form of caching, the consumer must always confirm with the producer whether the content is up to date. The validity of the cached copy is determined by some logic in the producer. The advantage of this approach is that the producer controls the use of the cached content rather than relying on a fixed period. For more information, see Section 61.3.7.1, "Activating Caching" and Section 61.3.7.3, "Adding Validation-Based Caching."

This section includes the following subsections:

- Section 61.3.7.1, "Activating Caching"
- Section 61.3.7.2, "Adding Expiry-Based Caching"
- Section 61.3.7.3, "Adding Validation-Based Caching"

Before You Begin

The steps that follow assume that you have:

- Followed through and understood Section 60.2.4, "How to Create a PDK-Java Portlet."

- Built a portlet using the wizard and successfully added it to a page.

61.3.7.1 Activating Caching

To use the caching features in your producers, you must first activate the middle tier cache. This cache is known as the PL/SQL Cache because it is the same cache used by mod_plsql, the Oracle HTTP Server plug-in that calls database procedures, and hence database producers, over HTTP.

Usually, your administrator is responsible for the configuration details of caching.

61.3.7.2 Adding Expiry-Based Caching

Expiry-based caching is a simple caching scheme to implement, and can be activated declaratively in your XML producer definition. You can set an expiry time for the output of any `ManagedRenderer` you use by setting its `pageExpires` property to the number of minutes you want the output to be cached for. For example, suppose you want portlet output to be cached for one minute.

To add expiry-based caching:

1. After you have used the Create Oracle PDK-Java Portlet wizard to build a portlet as described in Section 60.2.4, "How to Create a PDK-Java Portlet," edit the `provider.xml` file and set the `pageExpires` property tag of showPage to 1. This sets an expiry entry of one minute for the portlet.

   By default the wizard generates a standard and compressed tag for `showPage`. Expand the tag to include a subtag of `pageExpires`:

   ```
   <showPage class="oracle.portal.provider.v2.render.http.ResourceRenderer">
      <resourcePath>/htdocs/mycacheportlet/MyCachePortletShowPage.jsp
         </resourcePath>
      <pageExpires>1</pageExpires>
   </showPage>
   ```

2. Test that the portlet is cached for 1 minute by adding some JSP code to your show page. You can simply add the current time to your JSP.

   ```
   <%@page contentType="text/html; charset=windows-1252"
      import="oracle.portal.provider.v2.render.PortletRenderRequest"
      import="oracle.portal.provider.v2.http.HttpCommonConstants"
      import="java.util.Date"
      import="java.text.DateFormat"
   %>

   <%
    PortletRenderRequest pReq = (PortletRenderRequest)
      request.getAttribute(HttpCommonConstants.PORTLET_RENDER_REQUEST);
    DateFormat df = DateFormat.getDateTimeInstance(DateFormat.LONG,
      DateFormat.LONG,pReq.getLocale());
    String time = df.format(new Date());
   %>

   <P>Hello <%=pReq.getUser().getName() %>.</P>
   <P>This is the <b><i>Edit</i></b> render mode!</P>
   <P>This information is correct as of <%=time%>.</P>
   ```

   When viewing the portlet, you see that the time (including seconds) is constant for one minute. After the time has expired, the portlet displays the most current time and a new cache is set.

61.3.7.3 Adding Validation-Based Caching

Adding validation-based caching requires slightly more effort, but gives you explicit control over exactly which requests to your producer are cache hits. As an example, you may want to update the cache only when data within the portlet has changed. To implement this algorithm, you must override the `prepareResponse` method. The signature of the `BaseManagedRenderer.prepareResponse` method is:

```
public boolean prepareResponse(PortletRenderRequest pr)
                        throws PortletException,
                               PortletNotFoundException
```

In your version of `prepareResponse()`, do the following:

- Retrieve the cached version identifier set by the consumer in the render request by calling the `HttpPortletRendererUtil.getCachedVersion()` method:

  ```
  public static java.lang.String getCachedVersion
      (PortletRenderRequest request)
  ```

- If the portlet finds the previously cached version valid, then the appropriate header must be set by calling the `HttpPortletRendererUtil.useCachedVersion()` method. It also instructs the `RenderManager` that it is not necessary to call `renderBody()` to render the portlet body.

  ```
  public static void useCachedVersion(PortletRenderRequest request)
  ```

  Otherwise, use `HttpPortletRendererUtil.setCachedVersion()` to generate a new version of the portlet, which is cached. It also indicates to the consumer that the `renderBody()` method has to be called to regenerate the portlet content.

  ```
  public static void setCachedVersion(PortletRenderRequest request,
                                      java.lang.String version,
                                      int level)
                               throws java.lang.IllegalArgumentException
  ```

For validation-based caching, you need not update `provider.xml`. You can view the portlet by refreshing the page or adding the portlet to a page and updating the content. If the content has changed, then the portlet shows the new content. If the content has not changed, then a cached version of the portlet is displayed.

61.4 Testing Portlet Personalization

If you have implemented personalization for your portlet, then the Personalize link only appears on the portlet for authenticated users. Hence, to test the personalization of a portlet (the Personalize link), you must have some form of security implemented for the application consuming the portlet. For testing purposes, you may prefer to just configure the most basic authentication possible. For more information, see Section 69.12, "Configuring Basic Authentication for Testing Portlet Personalization."

61.5 Creating a Struts Portlet

Oracle PDK contains extensions to integrate Apache Struts applications. This section explains how to build a portlet from an existing struts application. You can also follow these steps to create a portlet that uses the Model View Controller paradigm. The PDK-Java extensions described in this section rely on Apache Struts 1.1.

You can use the Struts framework to enhance portlet behavior. For example, a Struts controller may be used for page flow within a portlet. A portlet renderer is used as a bridge between portlet render requests and Struts servlet requests

The following code shows a portion of the producer definition file (`provider.xml`):

```
...
<renderContainer>true</renderContainer>
    <renderCustomize>true</renderCustomize>
    <autoRedirect>true</autoRedirect>
    <contentType>text/html</contentType>
    <showPage class="oracle.portal.provider.v2.render.http.StrutsRenderer">
        <defaultAction>showCustomer.do</defaultAction>
    </showPage>
</renderer>
...
```

The `showPage` of the Struts portlet contains two important components:

- The renderer class (`oracle.portal.provider.v2.render.http.StrutsRenderer`) receives the render requests and forwards them to the Struts Action Servlet.
- The `defaultAction` tag defines the Struts action that is used by default when the portlet is called for the first time.

The PDK-Java enables you to easily develop a view (portlet view) of your Struts application. This view enforces a consistent look and feel of your Struts portlet using portal styles, and enables the end user to use the application within the portal.

To create a Struts portlet, you must use the PDK-Java Struts tags, which are extensions of the default Struts JSP tags. This development process is similar to that of creating a standalone Struts application. For portlets in a producer application to use the Struts framework, the Struts components must be part of the same web application.

To publish a part of an existing Struts application as a portlet, Oracle recommends that you first create a new view to serve as the portlet view of your application. This view uses existing objects (`Actions`, `ActionForm`, and so on) with a new mapping and new JSPs.

> **Note:** Although Oracle recommends that you create a portlet view of your application, you could alternatively replace your application's struts tags with PDK-Java struts tags. This approach enables your application to run both as a standalone struts application and a portlet.

In this example, you create a portlet that enables you to add a new entry to a blog. Figure 61–3 and Figure 61–4 show how you submit a blog and save a blog entry.

Figure 61–3 Submitting a Blog



Figure 61–4 Saving a Blog Entry



`prepareNewBlog` is a simple empty action that redirects the request to the `enterNewBlog.jsp` page. This page shows a form for submitting a new blog.

The corresponding entry in the `struts-config.xml` is:

```
<action path="/prepareNewBlog" scope="request"
    type="view.PrepareNewBlogAction" >
    <forward name="success" path="/view/enterNewBlog.jsp"/>
</action>
<action path="/saveNewBlog" name="blogForm" scope="request"
    type="view.SaveNewBlogAction" input"/view/enterNewBlog.jsp"  >
    <forward name="success" path="/view/newBlogConfirmation.jsp"/>
</action>
```

This section includes the following subsections:

- Section 61.5.1, "How to Create a New Flow and View to Host the Portlet Actions"
- Section 61.5.2, "How to Create the New JSPs"
- Section 61.5.3, "How to Create a Portlet"
- Section 61.5.4, "How to Extend the Portlet to Add Business Logic"
- Section 61.5.5, "How to Register the Producer"

61.5.1 How to Create a New Flow and View to Host the Portlet Actions

To create a new view, first create a new set of `ActionMappings` (page flow) that redirects the various actions and requests to Portal-specific JSPs.

```
<action path="/portal/prepareNewBlog" scope="request"
    type="view.PrepareNewBlogAction" >
    <forward name="success" path="/view/portal/enterNewBlog.jsp"/>
</action>
<action path="/portal/saveNewBlog" name="blogForm" scope="request"
    type="view.SaveNewBlogAction" input="/view/enterNewBlog.jsp"  >
    <forward name="success" path="/view/portal/newBlogConfirmation.jsp"/>
</action>
```

As you can see, only the path attributes are modified. The `FormBean` Action responsible for the application business logic remains unchanged.

61.5.2 How to Create the New JSPs

As specified in the previous step, the actions forward the request to new JSPs, which are responsible for rendering the portlet content. Your new portlet view JSPs can share the HTML with the standalone view, but ensure the portlet meets the following criteria:

- Uses styles that enforce a consistent look and feel with the rest of the page.
- Contains HTML code that is allowed in HTML table cells (that is, no `<html>`, `<body>`, and `<frame>` tags).
- Uses the PDK-Java version of the Struts HTML tag library. This renders URLs in the correct format to work in a portlet context.

The PDK Struts HTML tag library contains versions of the Struts HTML tags that can be used in portlet markup.

Note: You can register the Oracle PDK with Oracle JDeveloper so that you can drop the tags from the Oracle JDeveloper Components Palette. For more information, see the *Registering a Custom Tag Library in JDeveloper* section in the Oracle JDeveloper online Help.

61.5.3 How to Create a Portlet

You can create your Struts portlet either manually or by using the Create Oracle PDK-Java Portlet wizard. Although the wizard does not explicitly offer Struts support, you can use the wizard to build your Struts portlet.

To create a portlet:

1. In JDeveloper, open the Oracle PDK-Java Portlet wizard.

   For more information, see Section 60.2.4, "How to Create a PDK-Java Portlet."

2. On the View Modes page of the wizard, for the Show Page mode, select an **Implementation Style** of **Java Class**.
3. For the **Package Name**, enter `oracle.portal.provider.v2.render.http`.
4. For the **Class Name**, enter `StrutsRenderer`. This generates the skeleton of the portlet renderer class, `StrutsRenderer`.
5. As the `StrutsRenderer` is part of the PDK, you do not need this generated file. So, when you finish the wizard, you must delete the file generated by the wizard. To do so, click the file in the System Navigator window, then from the **File** menu, select **Erase from Disk** in JDeveloper.
6. Edit the `provider.xml` and change the following properties:

At the producer level, perform the following:

- If the Struts application uses sessions (for example, the form sysnchronizer token mechanism is used or `<actionInSession>` is set to true), then enable session handling:

  ```
  <session>true</session>
  ```

At the portlet level, perform the following:

- If you want users to always return to the same portlet state as when they left the container page, then you can configure the struts renderer to save the struts action in the struts context:

  ```
  <actionInSession>true</actionInSession>
  ```

  If you prefer that users always start from the beginning of the portlet when they return from outside the container page, then you should not save the struts action:

  ```
  <actionInSession>false</actionInSession>
  ```

  Note that this setting is the default behavior.

- Specify the first action to raise when the portlet is called. Use the following code:

  ```
  <showPage class="oracle.portal.provider.v2.render.http.StrutsRenderer">
  <defaultAction>/portal/prepareNewBlog.do</defaultAction>
  </showPage>
  ```

61.5.4 How to Extend the Portlet to Add Business Logic

In your application, you should add code specific to your environment, such as the user's information, personalization, and localization. To do so, you can create a new `Action` class that is only called in context, and handles all business logic.

61.5.5 How to Register the Producer

Now that your portlet is ready to be used by consumers, you must make it accessible by registering it. For information about how to register your PDK-Java portlet, see Section 62.4, "Registering and Viewing Your Portlet."

62

Testing and Deploying Your Portlets

This chapter explains how to test and deploy JSR 286 and Oracle PDK-Java portlets. In a development environment, you can test your portlets on Integrated WebLogic Server (WLS), which comes packaged with Oracle JDeveloper, and also deploy Portlet Producer applications to an Oracle WebLogic Managed Server instance residing outside JDeveloper.

This chapter includes the following sections:

- Section 62.1, "Introduction to Portlet Deployment Testing in a Development Environment"
- Section 62.2, "Testing a Portlet Producer Application on Integrated WebLogic Server"
- Section 62.3, "Deploying a Portlet Producer Application to an Oracle WebLogic Managed Server Instance"
- Section 62.4, "Registering and Viewing Your Portlet"

For information about testing and deploying other WebCenter Portal: Framework applications, see Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application".

For information about creating portlets, see Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge" and Chapter 60, "Creating Portlets with the Portlet Wizard".

62.1 Introduction to Portlet Deployment Testing in a Development Environment

Before you deploy your Portlet Producer application, you are advised to test it in a development environment. Integrated WLS in JDeveloper enables you to test your portlets in a single step, without creating a deployment profile. To learn the benefits of testing applications on Integrated WLS, see Integrated WebLogic Server. An additional benefit is that portlet customizations that you perform at design time are maintained in your application workspace and become an integral part of the application source definition. These changes are packaged with the EAR file when the application is deployed to a WebLogic Managed Server instance (managed servers). This eliminates the requirement to export and import portlet customizations.

If, while testing your application with Integrated WebLogic Server, you make changes at runtime, these customizations are also retained. If you do not want to keep these runtime customizations when you deploy your application to a managed server, you must clear them. You can do this manually by deleting them from the configured directory. Alternatively, during the development of your application, you can clear out these runtime customizations each time the application is run. To do this:

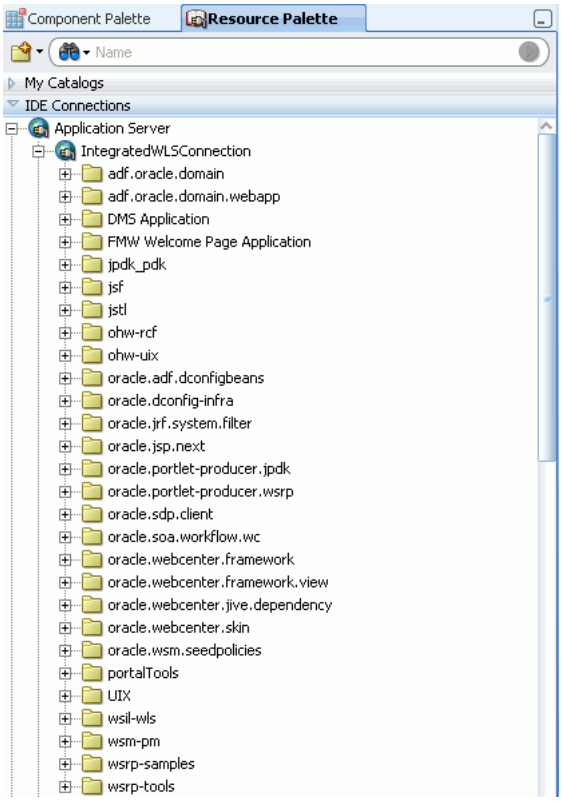
1. In the Application Navigator, click the **Application Menu** icon next to your application's name and choose **Application Properties**.
2. In the Application Properties dialog, expand the **Run** node in the left pane, then select **MDS**.
3. In the right pane, select **Delete customizations before each run**. Doing so clears the MDS of any runtime customizations every time the application is run.
4. Click **OK**.

To test your Portlet Producer application on an Oracle WebLogic Managed Server instance, or to deploy for production, you must configure a connection to the managed server, create deployment plans, and then deploy to the server instance as described in this chapter. For information about deploying Portlet Producer applications using Oracle Enterprise Manager Fusion Middleware Control, WLS Administration Control, or WLST commands, see the chapter, "Managing Portlet Producers" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

62.2 Testing a Portlet Producer Application on Integrated WebLogic Server

The connection to Integrated WLS (Default Server) connection is preconfigured and shows as **IntegratedWLSConnection** under the **Application Server** node in the **IDE Connections** panel of the Resource Palette, as shown in Figure 62–1. You can run multiple applications simultaneously and watch the progress of each application in the Run Manager panel. The Run Manager panel also lets you stop Default Server instances.

Figure 62–1 IntegratedWLSConnection in IDE Connections



For more information about Integrated WLS Server, see Integrated WebLogic Server.

This section includes the following subsections:

- Section 62.2.1, "How to Test JSR 286 Portlets on Integrated WebLogic Server"
- Section 62.2.2, "What Happens When You Test JSR 286 Portlets on Integrated WebLogic Server"
- Section 62.2.3, "How to Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server"
- Section 62.2.4, "What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server"

62.2.1 How to Test JSR 286 Portlets on Integrated WebLogic Server

To test a JSR 286 Portlet Producer application, right-click the `portlet.xml` page in the project folder and select **Run**. Running `portlet.xml` triggers packaging and deployment of your Portlet Producer application on an instance of Integrated WLS named after the application.

> **Note:** When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable.
>
> For a more persistent testing scenario, you can deploy your Portlet Producer application to Integrated WLS by right-clicking the application and then selecting **Deploy**. This deploys your application to the Default Server instance so that it is always available when the Default Server is running. Thus, the application remains accessible for multiple consumer applications. If you choose this method, then you must first create deployment profiles, as described in Section 62.3.1, "How to Create Deployment Profiles". If you deploy your application to Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the **Repository Name** field.

See the IntegratedWebLogicServer - Log window to monitor the deployment progress. The log shows the URL of the application page. The WSRP producer URL uses the following syntax:

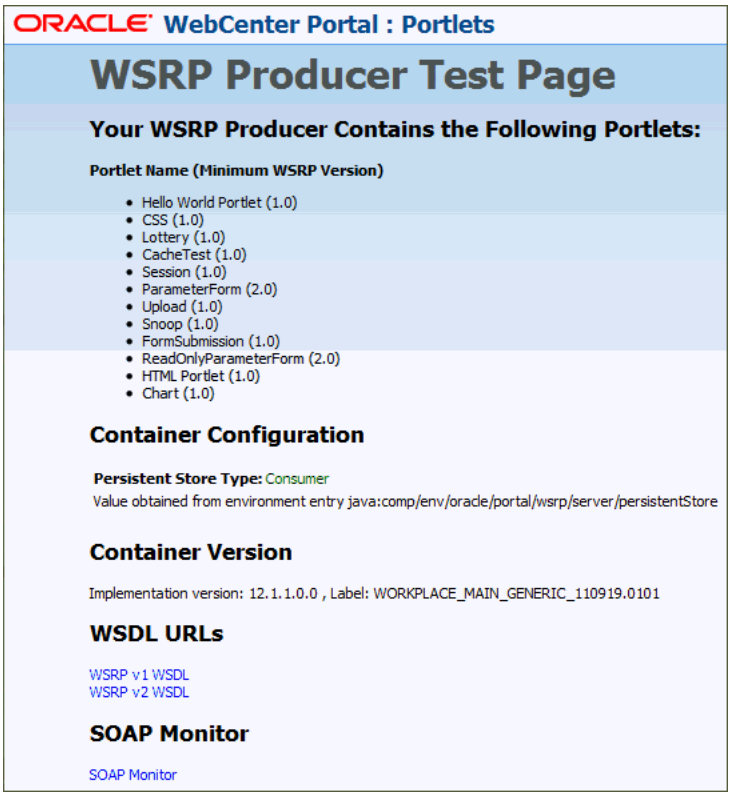
```
http://host:port/applicationname-Portlets-context-root/info
```

where:

- `host` is the server to which your producer has been deployed.
- `port` is the HTTP Listener port. Typically, it is `7101`. When the server is started, the port is displayed in the console.
- `context-root` is the Web application's context root.

A test page similar to Figure 62–2 displays in a browser window.

Figure 62–2 WSRP Producer Test Page



Note: This procedure is for testing purposes only. After this procedure, you are required to register your producer as described in Section 62.4, "Registering and Viewing Your Portlet".

62.2.2 What Happens When You Test JSR 286 Portlets on Integrated WebLogic Server

When you run JSR 286 portlets on Integrated WLS, the following happens:

- `wsdls` and other configuration files are added to the WEB-INF directory to configure the portlets as a Web service.
- The `web.xml` file is updated with listener and server classes, filters, parameters, and other configurations that are required to run the JSR 286 Portlet Producer application successfully.

  For example, the `oracle.portlet.server.adapter.web.ServerContextListener` class, `WSRP_v2_PortletManagement_Service` and `WSRPBaseService` filters, and so on.
- Libraries required for JSR 286 portlets are added to the `weblogic.xml` file, for example, `oracle.portlet-producer.wsrp`.

These configurations vary depending upon the portlet requirements.

62.2.3 How to Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server

When you run your Portlet Producer application on Integrated WLS, an instance of Default Server starts.

To test a Portlet Producer application, right-click a JSP page, for example, `index.jsp` in the project folder and select **Run**. Running `index.jsp` triggers packaging and deployment of your Portlet Producer application on an instance of Integrated WLS named after the application.

> **Note:** When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable.
>
> For a more persistent testing scenario, you can deploy your Portlet Producer application to Integrated WLS by right-clicking the application and then selecting **Deploy**. This deploys your application to the Default Server instance so that it is always available when the default server is running. Thus, the application remains accessible for multiple consumer applications. If you choose this method, then you must first create deployment profiles, as described in Section 62.3.1, "How to Create Deployment Profiles". If you deploy your application to Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the **Repository Name** field.

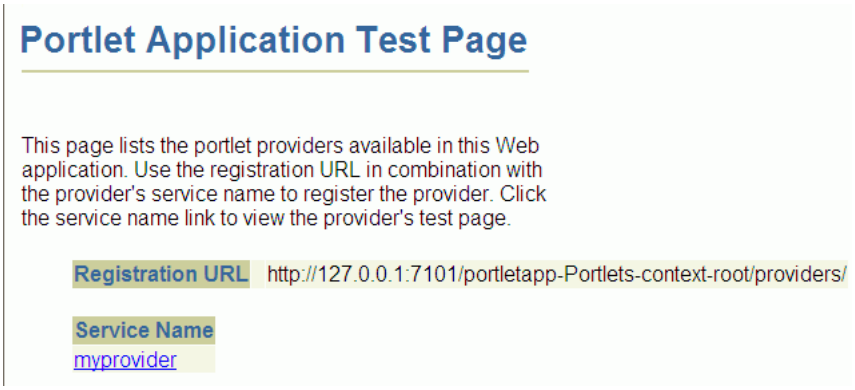
See the DefaultServer Log window to monitor the deployment progress. The DefaultServer - Log shows the URL of the application test page, as shown in Figure 62–3. The format of the URL is `http://`*`host:port/application_name`*`-Portlets-context-root/index.jsp`.

Figure 62–3 DefaultServer - Log



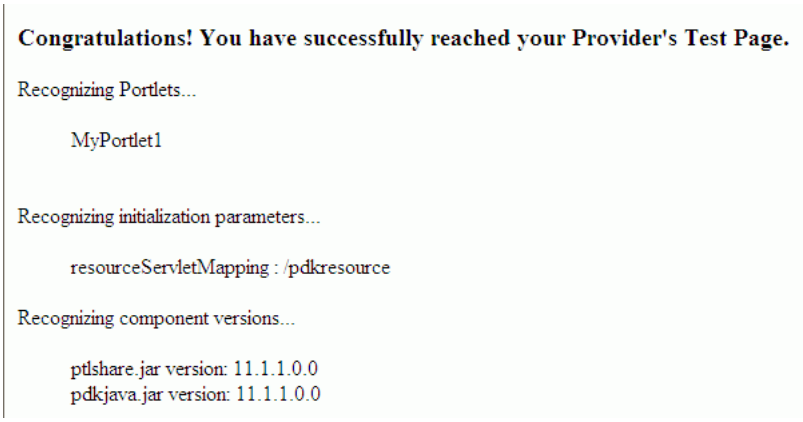
The PDK-Java Application Test page displays in a browser window, as shown in Figure 62–4.

Figure 62–4 Portlet Producer Application Test Page



Click the link underneath **Service Name**. Your browser should open with a page similar to the one shown in Figure 62–5. The URL of this page is the one required to register the producer with another application.

Figure 62–5 Producer Test Page



Alternatively, you can construct the URL yourself as follows:

`http://`*`host`*`:`*`port`*`/`*`context-root`*`/providers/`*`producer_name`*

where:

`host` is the server to which your producer has been deployed.

`port` is the HTTP Listener port. Typically, it is `7101`. When the server is started, the port is displayed in the console.

`context-root` is the Web Application's Context Root, which you specified earlier and can be found in the WAR Deployment Profile Properties under General.

`producer_name` is the name of the portlet's producer. A WAR file may contain multiple producers, hence you should always include the name of the producer for clarity. Otherwise, you get the default producer, which is the first producer created. The default producer is defined by the `_default.properties` file. This is created with the first producer in a project.

If you enter this URL in your browser, you should see a page similar to the one in Figure 62–5.

62.2.4 What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server

When you run PDK-Java portlets on Integrated WLS the following happens:

- Configuration files for PDK-Java portlets are added to the WEB-INF directory.
- The `web.xml` file is updated with listener and server classes, filters, parameters, and other configurations that are required to run the Portlet Producer application successfully.

  For example, the ResourceServlet pdkresource, the `oracle.portlet.server.service.ContextFilter` filter, and so on. These configurations vary depending upon the portlet requirements.
- Libraries required for PDK-Java portlets are added to the `weblogic.xml` file, for example, `oracle.portlet-producer.jpdk`.

62.3 Deploying a Portlet Producer Application to an Oracle WebLogic Managed Server Instance

To test your Portlet Producer application on an Oracle WebLogic Managed Server instance, or to deploy it for production, you must first create an application WAR deployment profile, a deployment descriptor, and a connection to the Oracle WebLogic Managed Server instance. Before you deploy or test your Portlet Producer application to a managed server, ensure that the managed server is created using the correct template and it contains all the required shared libraries as described in the section "Creating a Managed Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Note: Before you deploy the application, you must remove the `test-all` application role if you enabled the automatic grants feature in the Configure ADF Security wizard. Because the `test-all` role makes all ADF resources public, its presence increases the risk that your application may leave some resources unprotected. You must therefore remove the role before you migrate the application-level policy store.

For more information, see the section "How to Remove the test-all Role from the Application Policy Store" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

For information about deploying Portlet Producer applications using Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Administration Console, or WLST commands, see the chapter, "Managing Portlet Producers" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following subsections:

- Section 62.3.1, "How to Create Deployment Profiles"
- Section 62.3.2, "How to Create and Provision a WebLogic Managed Server Instance"
- Section 62.3.3, "How to Create and Register the Metadata Service Repository"
- Section 62.3.4, "How to Create a WebLogic Managed Server Connection"

- Section 62.3.5, "How to Deploy a Portlet Producer Application to an Oracle WebLogic Managed Server Instance"
- Section 62.3.6, "What Happens When You Deploy a Portlet Producer Application to an Oracle WebLogic-Managed Portlet Server"

62.3.1 How to Create Deployment Profiles

The project-level deployment profile is packaged as a Web Application Archive (WAR) file. Deployment descriptors are server configuration files that define the configuration of an application for deployment and are deployed with the Portlet Producer application as needed. The deployment descriptors that a project requires depend on the technologies the project uses and on the type of the target application server.

Note: You can deploy PDK-Java portlets only as EAR files. Therefore, while creating deployment profiles, ensure that the WAR file is included in the application's EAR file. For information about how to create an EAR file, see Section 70.3.2.2, "Creating Deployment Profiles" in Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application".

This section includes the following subsections:

- Section 62.3.1.1, "Creating a WAR Deployment Profile"
- Section 62.3.1.2, "Creating a Deployment Descriptor"

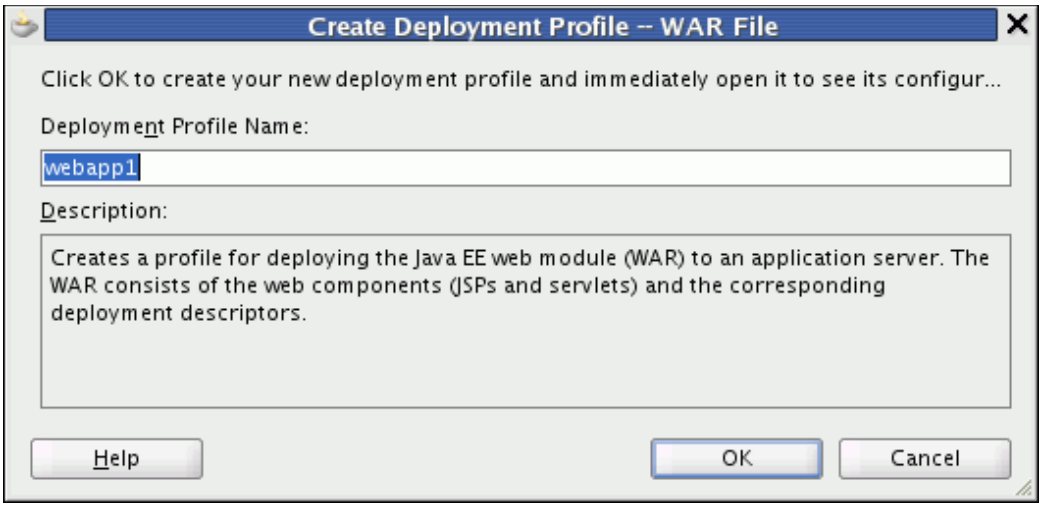
62.3.1.1 Creating a WAR Deployment Profile

To create a WAR deployment profile:

1. In the Application Navigator, expand **Web Content** and **WEB-INF**.
2. Right-click **web.xml** and choose **Create WAR Deployment Profile**.

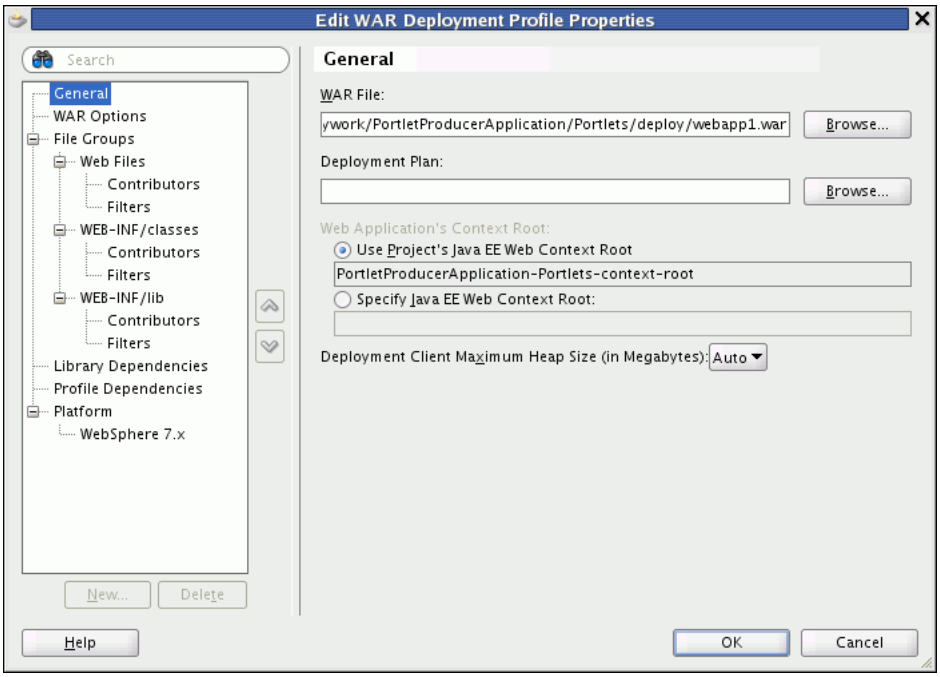
   Alternatively, from the main menu, choose **File** and then **New**. In the New Gallery, expand **General**, select **Deployment Profiles** and then **WAR File**, and click **OK**.
3. In the Create Deployment Profile -- WAR File dialog (Figure 62–6), enter a name for your deployment profile and click **OK**.

Figure 62–6 The Create Deployment Profile -- WAR File Dialog



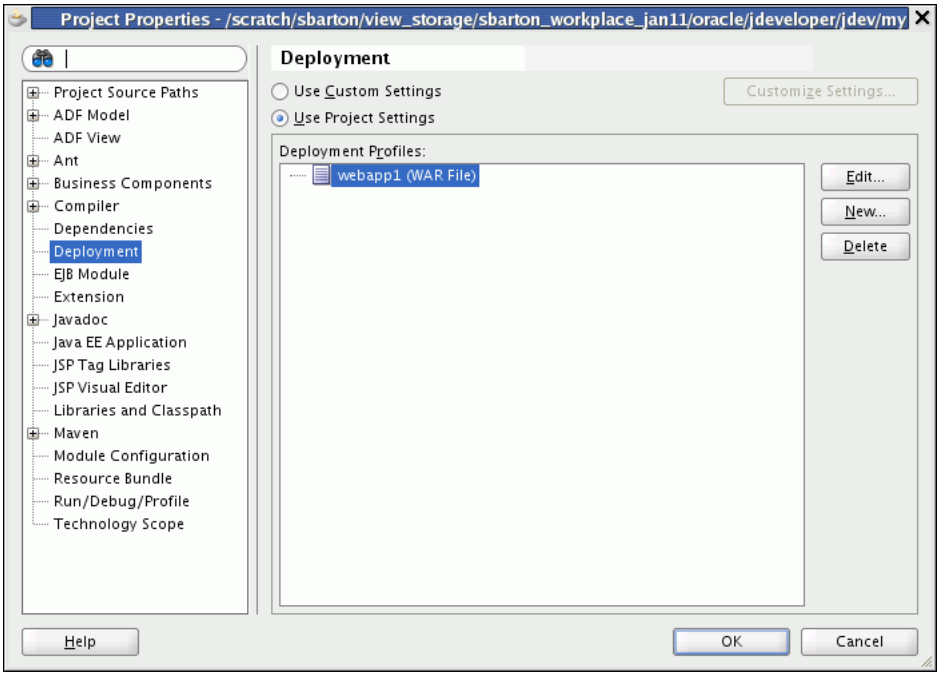
4. In the Edit WAR Deployment Profile Properties dialog (Figure 62–7), select the **Specify Java EE Web Context Root** option and enter a context root. Then click **OK**.

Figure 62–7 The Edit WAR Deployment Profile Properties Dialog



5. In the Project Properties dialog (Figure 62–8), under **Deployment Profiles**, select the WAR File you just created and click **OK**.

Figure 62–8 The Project Properties Dialog



62.3.1.2 Creating a Deployment Descriptor

To create a deployment descriptor:

1. From the main menu, choose **File** and then **New**.
2. In the New Gallery, expand **General**, select **Deployment Descriptors** and then **Java EE Deployment Descriptor**, and click **OK**.
3. In the Create Java EE Deployment Descriptor dialog, on steps 1 to 4, accept the default values and click **Finish**.

62.3.2 How to Create and Provision a WebLogic Managed Server Instance

To deploy a portlet producer application to a WebLogic Managed Server instance, you must first create a server instance that contains all the required shared libraries. For more information, see the section "Creating a Managed Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

62.3.3 How to Create and Register the Metadata Service Repository

After creating the WebLogic Managed Server instance, you must create and register a Metadata Service Repository (MDS) schema for the application on the WebLogic Domain's Administration Server instance. For more information, see the section "Creating and Registering the Metadata Service Repository" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

62.3.4 How to Create a WebLogic Managed Server Connection

In JDeveloper, you can deploy your Portlet Producer applications to Oracle WebLogic Managed Server instances that reside outside JDeveloper. You must first create a connection to the server instance.

Before You Begin

Before you create a connection to the managed server, ensure that the server instance is up and running.

To create a WebLogic Managed Server connection:

1. From the main menu, select **File** and then **New**.
2. In the New Gallery, expand **General**, select **Connections** and then **Application Server Connection**, and click **OK**.
3. In the Create Application Server Connection wizard, on Step 1, enter a name for the new connection, for example, `PortletServer`. Then click **Next**.
4. On Step 2, specify the user name and password for authentication and click **Next**.
5. On Step 3, enter the host name of the WebLogic Server instance, for example, `webcenter.portletserver.example.com` and the port number, for example, `7001`.
6. In the WLS Domain, specify the name of the domain in which the WebLogic Server instance is created, for example, `wc_domain`. Click **Next**.
7. On Step 4, click **Test Connection**. If the test is successful, the connection is set up.
8. Click **Finish**.

62.3.5 How to Deploy a Portlet Producer Application to an Oracle WebLogic Managed Server Instance

After you have created the deployment profiles and a connection to the managed server for portlet deployment, you can deploy your Portlet Producer application to this server instance.

To deploy a Portlet Producer application:

1. In the Application Navigator, open the application to be deployed.
2. Right-click the project folder, choose **Deploy**, and then the deployment profile.
3. In the Deploy dialog, in the Deployment Action page, select **Deploy to Application Server**, and then click **Next**.
4. In the Select Server page, select the connection to the managed server, and then click **Next**.
5. In the WebLogic Options page, select the managed server, for example, `WLS_Portlet`, and click **Finish**.

> **Note:** If your Portlet Producer application contains JSR 286 portlets, then the Select deployment type dialog displays. Click **Yes** to add the configuration required to expose this application as a WSRP service.

6. In the Deployment Configuration dialog, under the **MDS** tab, in the **Repository Name** dropdown list, select the metadata repository to be used by the application you are deploying.
7. In the **Partition Name** field, specify the application's partition name. Each application is recommended to have its own partition. The application name can be used as the partition name.

8. Under the **Connections** tab, modify the connections packaged with the portlet producer application being deployed.

   Note: Consider the following about producer connections:

   - In the **Connections** tab, you cannot change secure properties of the connections. You can modify these connections post-deployment using Fusion Middleware Control. For information, see the chapter "Managing Services" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
   - To edit a WSRP connection, you must also edit the associated Web service connection, which follows the naming convention *connectionname*`-wsconn`, for example, `myWSRPproducer-wsconn`.
   - To edit PDK-Java producers, you must also edit the underlying URL connection, which follows the naming convention *connectionname*`-urlconn`, for example, `myPDKproducer-urlconn`.
   - WSRP 1.0 standard does not support export and import capabilities. Therefore, customizations made to WSRP 1.0 producers are not migrated during deployment.

9. Click **Deploy**. The Deployment - Log displays the deployment status.

   The message **Deployment started** displays in the Deployment - Log window. If the application is successfully deployed to the targeted server instance, the message **Deployment finished** displays in the log.

62.3.6 What Happens When You Deploy a Portlet Producer Application to an Oracle WebLogic-Managed Portlet Server

If you are deploying JSR 286 portlets, the configuration settings described in Section 62.2.2, "What Happens When You Test JSR 286 Portlets on Integrated WebLogic Server" are added to the EAR file.

If you are deploying PDK-Java portlets, the configuration settings described in Section 62.2.4, "What Happens When You Test PDK-Java Portlet Producer Applications on Integrated WebLogic Server" are added to the application EAR file at design time.

If you are deploying your Portlet Producer application in a cluster environment, you will receive the following warning:

```
WARNING: oracle.webcenter.lifecycle.exception.LifecycleLockedException: A lock
exists that prevents the export set import.
```

This is expected behavior. When you deploy your application, the producer metadata exported into the EAR file (as a MAR file) needs to be imported into an MDS schema for use in the production environment. Importing the metadata occurs automatically during deployment. This import only needs to happen with one node in the cluster, so a lock is created to prevent other nodes in the cluster trying to perform the same operation.

62.4 Registering and Viewing Your Portlet

After you have created and deployed the producer and its portlets, you should register the producer with an application and add one or more portlets to a page to check that it is working correctly. Registering a producer gives applications the information they require to locate and communicate with that producer. After you register a producer, it is exposed as a connection, and the producer and its portlets become available in the Application Resources panel under the Connections node, or in the Resource Palette.

To register producers for your JSR 286 portlets, follow the instructions provided in Section 64.2.1, "How to Register a WSRP Portlet Producer".

To register producers for your PDK-Java portlets, follow the instructions provided in Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer".

To add your portlets to a page, follow the instructions provided in Section 64.3, "Adding Portlets to a Page".

63

Creating Pagelets with Oracle WebCenter Portal's Pagelet Producer

A pagelet is a reusable user interface component. Any HTML fragment can be a pagelet, but pagelet developers can also write pagelets that are parameterized and configurable, to dynamically interact with other pagelets, and respond to user input. Pagelets are similar to portlets, but while portlets were designed specifically for portals, pagelets can be run on any web page, including within a portal or other web application. Pagelets can be used to expose platform-specific portlets in other web environments.

Oracle WebCenter Portal's Pagelet Producer (previously known as Oracle WebCenter Ensemble) provides a collection of useful tools and features that facilitate dynamic pagelet development. This chapter includes detailed information about pagelet development and configuration using the Pagelet Producer.

- Section 63.1, "About Server Communication and the Proxy": This section explains how the Pagelet Producer acts as a proxy server, brokering transactions between client computers and external resources.
- Section 63.2, "Using Pagelets in Web Applications": This section explains how to use pagelets in your WebCenter Portal applications.
  - Section 63.2.1, "Adding a Pagelet to a JSF Page in Oracle JDeveloper": This section explains how to use Oracle JDeveloper to drag and drop pagelets onto a JSF page.
  - Section 63.2.2, "Adding a Pagelet to a Web Page": This section explains how to insert a pagelet into a proxied or non-proxied page using JavaScript or REST.
  - Section 63.2.3, "Adding a Pagelet to a Page in WebCenter Portal: Spaces": This section explains how to add a pagelet to a space using Composer.
- Section 63.3, "Building Pagelets Using the Pagelet Producer": This section provides detailed information about building pagelets using the Pagelet Producer.
  - Section 63.3.1, "Using the Adaptive Pagelet Scripting Framework": This section explains how the adaptive pagelet scripting framework can be used to create a coordinated page with dynamic, interactive functionality.
  - Section 63.3.2, "Using Pagelet Producer Credential Mapping": This section explains how to configure security settings for pagelets and resources to manage authentication for users.
  - Section 63.3.3, "Modifying Pagelet Functionality at Runtime": This section explains how to use custom web injectors and parsers to modify pagelet functionality at runtime.

  - Section 63.3.4, "Debugging Pagelets": This section explains how to access advanced logging traces for debugging pagelets.
- Section 63.4, "Working with WSRP and Oracle JPDK Portlets": This section explains how the Pagelet Producer can be used to present WSRP and Oracle JPDK portlets for use in any web application.
  - Section 63.4.1, "Using Pagelet Chrome": This section explains how to use pagelet chrome to modify markup at runtime in WSRP or Oracle JPDK portlets.
- Section 63.5, "Working with OpenSocial Gadgets": This section summarizes Pagelet Producer support for standard OpenSocial APIs and explains how to use gadgets to store preferences and retrieve user information from WebCenter Portal using OpenSocial APIs.
  - Section 63.5.1, "Configuring Authentication"
  - Section 63.5.2, "Storing User Preferences"
  - Section 63.5.3, "Accessing WebCenter Portal Profile Information"
  - Section 63.5.4, "Accessing a User's Activity Stream"
  - Section 63.5.5, "Using Gadget Eventing"
  - Section 63.5.6, "Example: Consuming an External OpenSocial Gadget"
  - Section 63.5.7, "Example: Consuming a Local OpenSocial Gadget"

63.1 About Server Communication and the Proxy

The Pagelet Producer acts as a proxy server, brokering transactions between client computers and external resources.

Services on external resources communicate with the Pagelet Producer via HTTP. For example, when a browser requests a page, the Pagelet Producer makes simultaneous requests to each external resource to retrieve the pagelet content for the page. The external resource reads the current user's preferences from the HTTP headers sent by the Pagelet Producer and sends back the appropriate HTML. The Pagelet Producer inserts the HTML into the page markup. Any images stored in the Image Service are retrieved and displayed by the browser.

63.1.1 The Pagelet Producer Proxy

A proxy server acts as a middleman, brokering transactions between a client computer and another server. This configuration is typically used to serve content to clients that would otherwise be unable to access the external resource, but it can be used to impose additional security restrictions on the client, including the use of policies. The proxy hides the external resource; to the end user, the content appears to come directly from the proxy server.

This architecture makes the Pagelet Producer the single point of access for content, and allows external resources to reside on a private network or behind a firewall. As long as the Pagelet Producer can connect to the external resource, users can view the content, even if they cannot access it directly. To the browser, the Pagelet Producer appears to be the source of content on the external resource.

When a user interacts with a service, any request made to a URL in the proxy is automatically rerouted through the Pagelet Producer. To the user, the content appears

to come from the Pagelet Producer; the external resource is an unknown back-end system.

There are many benefits to this configuration. The most useful to services are:

- **Dynamic functionality and personalization**: The Pagelet Producer intercepts requests from pagelets, which allows it to include information stored in the MDS in HTTP requests and responses.
- **Security**: Services can allow users to access content that is not publicly available. Files stored on a secure server can be made available by including specific URLs in the configuration of the proxy. Note: The proxy is a powerful feature, and can compromise security if incorrectly configured. Allowing direct access to a external resource that hosts unprotected private content could create a dangerous security hole.
- **Performance**: The Pagelet Producer caches proxied content, decreasing response time for end users and improving performance on the external resource. While proxying works efficiently for content like HTML, it is generally not appropriate for binary data like static images. Images do not need to be transformed, and proxying large images can adversely affect performance. This is one reason the Image Service should be used to prevent routing static images through the proxy.

The collection of URLs that should be proxied for a service is configured in the Resource editor. All URLs that use the Internal URL prefix configured for the resource or point to other proxied CSP or web resources will be proxied unless **Enable URL Rewriting** is deselected.

Keep the following warnings and best practices in mind when implementing services that use the proxy:

- **URL transformation**: The Pagelet Producer must transform code so that proxied URLs open correctly. Before the Pagelet Producer sends a response, it parses the HTML and looks for any URLs that use the Internal URL prefix configured for the associated producer resource. The Pagelet Producer transforms any URLs that should be proxied before returning the response to the client. Relative URLs are transformed to point to the correct location.
- **Scripting limitations**: JavaScript constructs that dynamically create URLs can cause problems, because they are run after content is already transformed. VBScript is not transformed by the proxy; you can continue to use dynamic scripts and VBScript as long as your code is proxy-aware. Parsers and web injectors can be used to implement targeted control over URL rewriting. For details on configuring parsers and web injectors, see the *Managing Oracle WebCenter Portal's Pagelet Producer* chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- **URL encoding**: It is a best practice to encode all headers that are URLs to prevent unexpected transformation. In JSP, encode all URLs that are written. If the code writes URLs in the body of a page (for example, a link to a preferences page) it should be encoded. The standard Java servlet command response.encodeURL() is the preferred method, but you can also use URLEncoder.encode(url). In the .NET Framework, the HttpUtility.URLEncode class provides the necessary functionality. Note: In .NET, there is no need to encode the redirect URL; this is handled automatically on the back end.

63.1.1.1 About Pagelets and the Proxy

All pagelets are designed to be displayed with other pagelets. As explained in the previous section, the Pagelet Producer acts as a proxy, processing and combining

pagelets from multiple applications to create a single, unified page with a range of functionality.

The code returned by a pagelet is parsed by the Pagelet Producer server and inserted into the HTML markup that makes up the page. Pagelets from the same back-end application can interact with each other within the page.

63.1.2 About HTTP and CSP

HTTP is a protocol used mostly for transferring web page content and XML between a server and a client. CSP is a platform-independent protocol based on the open standard of HTTP 1.1 that defines the syntax of communication between the Pagelet Producer and external CSP resources.

63.1.2.1 HTTP

HTTP communication is made up of Requests and Responses. Requests and Responses are essentially lists of name-value pairs of metadata in headers, along with an optional body. The body is the data that is being transferred (an HTML page or XML file). The metadata in the headers is information about the Request or Response itself (what language the content is in, or how long the browser should cache it). The Request and Response each contain specific information, outlined next. For more detailed information on HTTP, see RFC 2616 (http://www.faqs.org/rfcs/rfc2616.html).

The client sends the server an **HTTP Request**, asking for content. The Request body is used only for requests that transfer data to the server, such as POST and PUT.

HTTP Request Format:

```
[METHOD] [REQUEST-URI] HTTP/[VERSION]
[fieldname1]: [field-value1]
[fieldname2]: [field-value2]
[request body, if any]
```

HTTP Request Example:

```
GET /index.html HTTP/1.1
Host: www.plumtree.com
User-Agent: Mozilla/3.0 (compatible; Opera/3.0; Windows 95/NT4)
Accept: */*
Cookie: username=JoeSmith
```

The server sends back an **HTTP Response** that contains page content and important details, such as the content type, when the document was last modified, and the server type. The Response contains an error message if the requested content is not found.

HTTP Response Format:

```
HTTP/[VERSION] [CODE] [TEXT]
[fieldname1]: [field-value1]
[fieldname2]: [field-value2]
[response body, if any (document content here)]
```

HTTP Response Example:

```
HTTP/1.0 200 Found
Last-modified: Thursday, 20-Nov-97 10:44:53
Content-length: 6372
Content-type: text/html
<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 3.2 Final// EN'><HTML>
...followed by document content...
```

Custom HTTP headers can be configured to include specialized information.

> **Note:** Header size limits are controlled by the server that hosts the code. The standard limit for IIS/ASP is 60K. Java Application Servers range from 2K to 10K. These limits are generally configurable; see your server documentation for details.

Services can also access standard HTTP headers, such as the Set-Cookie header or HTTP 1.1 basic authentication header. If you want to investigate HTTP further, you can view all the headers being passed back and forth between your browser and Web server using logging (as described in Section 63.3.4, "Debugging Pagelets"). HTTP is used in conjunction with SSL to serve up secure content. Single Sign-On (SSO) also uses HTTP headers for basic authentication.

63.1.2.2 CSP

CSP extends HTTP and defines proprietary headers to pass settings between Pagelet Producer and external CSP resources (i.e., Oracle WebCenter Interaction portlets). CSP outlines how these services use HTTP to communicate and modify settings. The latest version is 1.4, and is available here: http://download.oracle.com/docs/cd/E13158_01/alui/wci/docs103/devguide/references/CSP1.4.pdf.

The custom CSP headers used to communicate system and user configuration variables can also be used by pagelets.

Table 63–1 Pagelet Producer ***Headers***

Header Name	Description
User ID	The User ID of the currently logged in user. This value can be used to determine if the session has expired. If UserID=2, the default 'Guest' user is logged in; any other user's session has ended.
User Name	The name of the logged in user. The user's name can be used to personalize display or pre-fill form fields.
Image Service URL	The URL to the root virtual directory of the Image Service in the user's implementation of the Pagelet Producer. This location should be used for all static images used in services.
Stylesheet URL	The URL to the current user's style sheet. In each implementation of the Pagelet Producer, the UI is customized. In some portals, users can choose between a selection of stylesheets. Using these styles ensures that pagelets appear in the style of the current user's implementation of the Pagelet Producer.
Pagelet ID	The ID for the current resource (pagelet), and the instance ID for the current pagelet. This value is useful for appending to the names of HTML forms and client-side JavaScript functions to ensure unique form and function names on the page to avoid name conflicts.
Host Page URL	The URL to the page that hosts the pagelet. Preference pages need this URL to return the user to the correct page after settings are configured. This value can also be used to allow a single pagelet to display different content on different pages.

63.2 Using Pagelets in Web Applications

This section describes how to add pagelets to a JSF page in JDeveloper, to any web page, or to a page in WebCenter Portal: Spaces. Before you can add a pagelet to any web application, you must deploy and configure the resource and pagelet in the Pagelet Producer as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following subsections:

- Section 63.2.1, "Adding a Pagelet to a JSF Page in Oracle JDeveloper"
- Section 63.2.2, "Adding a Pagelet to a Web Page"
- Section 63.2.3, "Adding a Pagelet to a Page in WebCenter Portal: Spaces"

63.2.1 Adding a Pagelet to a JSF Page in Oracle JDeveloper

Oracle JDeveloper allows you to drag and drop pagelets onto a JSF page. In this section, you will register the Pagelet Producer with your WebCenter Portal: Framework application, add a pagelet to a JSF page, and view the page in a browser to test the pagelet.

This section includes the following subsections:

- Section 63.2.1.1, "Registering the Pagelet Producer with a WebCenter Portal: Framework Application"
- Section 63.2.1.2, "Adding a Pagelet to a JSF Page"
- Section 63.2.1.3, "Securing a Pagelet"

63.2.1.1 Registering the Pagelet Producer with a WebCenter Portal: Framework Application

Before you can add a pagelet to a JSF page, you must register the Pagelet Producer with your application. You can register a Pagelet Producer in two ways:

- Register a producer with a specific application. By default, the IDE Connection option is selected in the New Pagelet Producer dialog that creates the connection under the Application Resources panel.
- Register a producer using the Resource Palette. This option enables you to use the producer's pagelets in multiple applications. A pagelet that is available in the Resource Palette can be added to any of your Framework applications by dropping it on the page. When you add a pagelet from the Resource Palette, its producer gets registered with the application if that producer is not already registered with the application. You can drag and drop a whole producer connection from the Resource Palette into the Application Resources panel of the Application Navigator. This registers the producer with the application. Alternatively, you can right-click a producer in the Resource Palette and choose **Add to Application** from the context menu to register the producer with the currently open application

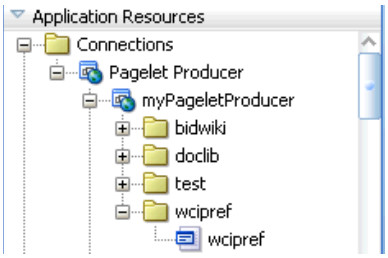
To register a Pagelet Producer through Oracle JDeveloper:

1. In the Application Resources panel of the Application Navigator, right-click **Connections,** choose **New Connection** and then choose **Pagelet Producer.**

   For other methods of invoking the wizard, such as from the Resource Palette, see Section 1.6.2, "How Do I Access the Connection Wizards?"

2. In the **Name** field of the New Pagelet Producer dialog, enter a meaningful name for your producer. For example, `myPageletProducer`.
3. In the **URL** field, enter the URL of your Pagelet Producer in the format: `http://hostname:portnumber/pagelets`.
4. Click **OK**. In the Application Resources panel, the Pagelet Producer is created under the Pagelet Producer directory in the Connections directory, as shown in Figure 63–1.

Figure 63–1 Pagelet Producer in the Application Resources Panel

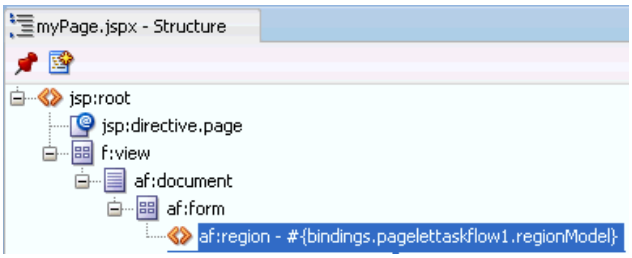


63.2.1.2 Adding a Pagelet to a JSF Page

To add a pagelet to a page:

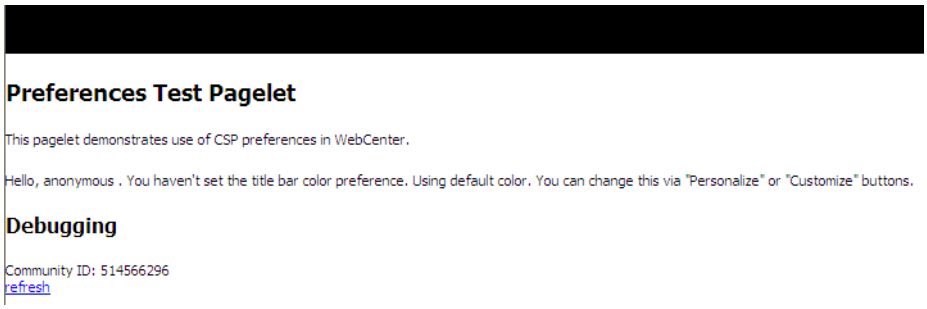
1. In JDeveloper, open your JSF page in Design View, if it is not open already.
2. Go to the Application Resources panel or the IDE Resources panel in the Resource Palette. Under **Pagelet Producer**, expand your producer to display its contents.
3. From the producer's contents, drop a pagelet onto the JSF page.
4. In the **Add Pagelet to Page** dialog, choose **Yes** in the Use IFrame section if you would like to enable IFRAME for this pagelet.
5. In the **IFrame Height** field, specify the required height in pixels or leave it blank. You can also specify the height as "auto" to use automatic resizing. (To use this option, you must add the IFrame resizing page to the project. For details, see Section 63.2.2.3, "Using Automatic Resizing with IFrames.") Click **OK**.
6. In the **Edit Task Flow Binding** dialog input parameters are specified by default. Change any parameters, if required, and then click **OK**.
7. Save the page. In the Structure window, the page looks like Figure 63–2.

Figure 63–2 Pagelet in the Structure Window



8. In the Application Navigator, right-click your JSF page under Projects, and choose **Run**.
9. If presented with a login page, enter user name and password so you can view the pagelet. Figure 63–3 shows a sample pagelet in a browser window.

Figure 63–3 Sample Pagelet



63.2.1.3 Securing a Pagelet

For information about securing pagelets, see the section "Policy" and the section "Configuring Oracle Single Sign-On (OSSO)" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

63.2.2 Adding a Pagelet to a Web Page

Once you have deployed a pagelet, you can insert it into a proxied or non-proxied page using JavaScript or REST.

This section includes the following subsections:

- Section 63.2.2.1, "Inserting Pagelets Using Javascript": The Pagelet Producer allows you to insert pagelets into non-proxied pages using a simple JavaScript function.
- Section 63.2.2.2, "Accessing Pagelets Using REST": Pagelet Producer REST APIs allow remote web services to retrieve information about resources and pagelets from the Pagelet Producer, and inject pagelets into proxied and non-proxied pages.
- Section 63.2.2.3, "Using Automatic Resizing with IFrames": The pagelet inject function can automatically resize the IFrame that encapsulates pagelet content.

63.2.2.1 Inserting Pagelets Using Javascript

You can insert pagelets into non-proxied pages using a simple JavaScript function.

To activate this feature, add the following HTML snippet in the <HEAD> section of the page.

```
<script type="text/javascript" src="http://proxy:port/pagelets/inject/v2/csapi">
</script>
```

This script injects all CSAPI and pagelet inject functions into the page to display the pagelet. One of the sections injected is the following function:

```
function injectpagelet(library, name, iframe_options, payload, params, context_id,
element_id,  is_in_community, chrome, forward_params)
{
    ...
}
```

This function injects a pagelet as a widget into the parent page. The method interface uses the following parameters:

- **library**: Required. A string representing the library name of the pagelet to inject. Accepts Unicode letters and numbers only; no spaces.

- **name**: Required. A string representing the name of the pagelet to inject. Accepts Unicode letters and numbers only; no spaces.
- **iframe_options**: Specifies whether to use IFRAME around the pagelet content. Sample IFRAME options: `iframe width=100% height=auto frameborder=0`. If omitted or left blank, the pagelet content is rendered inline.
- **payload**: The XML payload to send with the pagelet request.
- **params**: The pagelet parameters in query string format. For example, `'param1=value1&param2=value2&param3=value3'`.
- **context_id**: The external identifier of the pagelet instance, used to scope preferences on the Pagelet Producer server. Must be an integer.
- **element_id**: The HTML element ID in which the pagelet content is injected. If omitted or left blank, the pagelet content is injected using `document.write()` when the `injectpagelet` call is evaluated.
- **is_in_community**: Specifies whether this pagelet is on a community or group page. If the value is set to `true`, it sends the `context_id` in the community ID header to the pagelet. Defaults to `false`.
- **chrome**: Specifies the name of the chrome template to use for WSRP/JPDK pagelets. To suppress chrome, use a value of `none`.
- **forward_params**: Specifies whether to forward query string arguments from the consuming page to the back-end server. To suppress this functionality, use a value of `false`.

> **Note:** These arguments are positional; they must be provided in the given order. If you do not want to specify a particular argument, but do want to specify an argument that follows it, you must pass in an empty value ('') for the former. All arguments are optional except for library and name.

The script also creates a new `<div>` with a unique name that includes a reference to the `injectpagelet` function. Several examples are shown below:

```
<div>
    <script type="text/javascript">
        injectpagelet('library', 'name');
    </script>
</div>
<div>
    <script type="text/javascript">
    injectpagelet('library', 'name', 'iframe', 'payload',
'param1=value1&param2=value2&param3=value3');
    </script>
</div>

<div>
    <script type="text/javascript">
    injectpagelet('library', 'name', 'iframe width=100% height=200', 'payload');
    </script>
</div>
```

63.2.2.1.1 Adding a Preference Editor Using Javascript The `injecteditor` function lets you add preference editors that enable users to set personal and shared preferences for

pagelets that support this capability. This functionality is analogous to personalization and customization functionality in Oracle WebCenter Portal.

```
injecteditor(library, name, type, iframe_options, context_id, element_id, is_in_
community, chrome)
```

where:

- **library**: Required. The library name of the pagelet to inject. Accepts Unicode letters and numbers only; no spaces.
- **name**: Required. The name of the pagelet to inject. Accepts Unicode letters and numbers only; no spaces.
- **type**: The editor type. This argument supports these values: 'admin', 'pagelet', 'community'. In case of the 'community' argument, `context_id` is sent to pagelet in the community ID CSP header.
- **iframe_options**: Specifies whether to use IFRAME around the pagelet editor content. Sample IFRAME options: `iframe width=100% height=auto frameborder=0`. If omitted or left blank, the editor content is rendered inline.
- **context_id**: The external identifier of the pagelet instance, used to scope preferences on the Pagelet Producer server. Must be an integer.
- **element_id**: The HTML element ID in which the pagelet content is injected. If omitted or left blank, the pagelet content is injected using `document.write()` when the `injecteditor` call is evaluated.
- **is_in_community**: Specifies whether this pagelet is on a community or group page. If the value is set to `true`, it sends the `context_id` in the community ID header to the pagelet. Defaults to `false`.
- **chrome**: Specifies the name of the chrome template to use for WSRP/JPDK pagelets. To suppress chrome, use a value of `none`.

> **Note:** These arguments are positional; they must be provided in the given order. If you do not want to specify a particular argument, but do want to specify an argument that follows it, you must pass in an empty value ('') for the former. All arguments are optional except for library and name.

63.2.2.2 Accessing Pagelets Using REST

REST stands for Representational State Transfer and is a simple way of providing APIs over HTTP. The basic principles of REST are:

- API URLs point to the resource rather than a generic method endpoint.
- Requests use standard HTTP verbs for simplified CRUD methods. This is a read-only API and allows GET requests only.
- Every request returns a full representation of the object retrieved (pagelet or resource).

Pagelet Producer REST APIs provide the following functionality:

- Inject pagelets into non-proxied pages, allowing the Pagelet Producer to act as a portlet provider for Oracle WebCenter Interaction, Oracle WebLogic Portal, or other third-party portals. For details, see Section 63.2.2.2.1, "Pagelet Inject API".

- Allow remote web services to retrieve information about resources and pagelets from the Pagelet Producer. For details, see Section 63.2.2.2.2, "Data Retrieval APIs".

63.2.2.2.1 Pagelet Inject API The pagelet inject URL can be used in portals to specify the location of a remote portlet (this is how a pagelet can be used as a portlet). The inject URL can also be used as the src attribute of an IFrame tag in any HTML page.

The URL must use the following format:

`http://host:port/pagelets/inject/v2/pagelet/libraryname/pageletname?content-type=html`

where `libraryname` and `pageletname` refer to the library and pagelet configured in the Pagelet Producer.

> **Note:** When using the pagelet inject API as the URL for a Portlet Web Service in Oracle WebCenter Interaction, you must switch "pagelet" to "portlet" in the URL. For example, the above URL would become:
>
> `http://host:port/pagelets/inject/v2/`**`portlet`**`/libraryname/pageletname?content-type=html`

The query string arguments to the above call define how the pagelet is to be returned. The following parameters are defined:

- **instanceid:** Optional. The instance ID of the pagelet, used to uniquely identify the pagelet on the page to facilitate inter-pagelet communication. Must be unique to the page.
- **context**: Optional. The external identifier of the pagelet instance, used for scoping preferences on the Pagelet Producer server. Must be an integer.
- **content-type**: The return type. Three types are supported:
  - **javascript**: Returns injectable code.
  - **html**: Returns the pagelet markup with its associated PTPortlet object.
  - **iframe**: Returns an IFrame that points back to the inject api, filling the IFrame with the pagelet content, instead of directly inline with the page. The IFrame can be styled by providing a set of query string parameters.

Parameter	Description	Default
ifwidth	Sets the width of the IFrame; can be specified in percent '%' or pixels 'px', for example: ifwidth=500px. Can be set to 'auto' to automatically resize the IFrame to fit the content within. For details, see Section 63.2.2.3, "Using Automatic Resizing with IFrames".	100%
ifheight	Sets the height of the IFrame; can be specified in percent '%' or pixels 'px', for example: ifheight=500px. Can be set to 'auto' to automatically resize the IFrame to fit the content within. For details, see Section 63.2.2.3, "Using Automatic Resizing with IFrames".	No default
ifborder	Sets the border of the IFrame.	'none'

Parameter	Description	Default
ifalign	Sets the align rule within the IFrame, for example: ifalign=center.	No default
ifdesc	Sets the description of the IFrame.	No default
ifmarginheight	Sets the margin height; can be specified in percent '%' or pixels 'px', for example: ifmarginheight=500px.	No default
ifmarginwidth	Sets the margin width; can be specified in percent '%' or pixels 'px', for example: ifmarginwidth=500px.	No default
ifscrolling	Sets the scrollbars of the IFrame. Accepted values: yes/no/auto.	auto
ifstyle	Sets the CSS style of the IFrame	No default
ifclass	Sets the CSS class of the IFrame.	No default

- **csapi**: Optional. Sets whether the CSAPI will be included with the pagelet response (true or false). Including the CSAPI is optional, but the pagelet included in the response relies on the CSAPI libraries being present on the page where the pagelet is to be rendered. If csapi=false, then the CSAPI libraries must be included with the parent page (usually in the HEAD section).
- **onhtttperror:**: Optional. When a pagelet request results in a 403, 404 or any other error code, the Pagelet Producer can forward the error code and the error page itself to the browser for display to the user. The onhttperror parameter accepts the following values:
  - **comment** (default): The Pagelet Producer will create an HTML comment in place of the failing pagelet (the failing pagelet will simply not be displayed).
  - **inline**: The pagelet error along with the server error page will be displayed inline where the pagelet would normally be shown on the page.
  - **fullpage**: The http error will consume the whole page. This mode is only available if the Pagelet Producer controls the parent page.

For example, the following URL points to the linkspagelet in the samples library:

```
http://host:port/pagelets/inject/v2/pagelet/samples/linkspagelet
?content-type=iframe&csapi=true&ifheight=123px&ifclass=myclass
```

This URL should result in markup similar to the code below.

> **Note:** The IFrame source points back to the inject API, but this time the content-type parameter is set to `html`. This feature adds an additional step in the pagelet retrieval. The `csapi` parameter is set to `true` on the subsequent call to get the IFrame contents so that the required CSAPI content is included in the IFrame (if this was not the case, JavaScript resolve errors would be returned because the pagelet code cannot access any CSAPI script included outside the IFrame).

```
<html>
 <head>
 </head>
 <body>
  <iframe frameborder="none" class="myclass" width="100%" height="123px"
```

```
scrolling="auto"
src="http://proxy:port/inject/v2/pagelet/samples/linkspagelet?asdg=asdfgas&param=t
rue&content-type=html&jswrap=false&csapi=true">
   <html>
    <head>
     <script
src="http://proxy:loginserverport/loginserver/ensemblestatic/imageserver/plumtree/
common/private/js/jsutil/LATEST/PTUtil.js" type="text/javascript"> </script>
     <script
src="http://proxy:loginserverport/loginserver/ensemblestatic/imageserver/plumtree/
common/private/js/jsutil/LATEST/PTDateFormats.js" type="text/javascript"></script>
     <script
src="http://proxy:loginserverport/loginserver/ensemblestatic/imageserver/plumtree/
common/private/js/jsxml/LATEST/PTXML.js" type="text/javascript"></script>
     <script
src="http://proxy:loginserverport/loginserver/ensemblestatic/imageserver/plumtree/
common/private/js/jsportlet/LATEST/PTPortletServices.js"
type="text/javascript"></script>
    </head>

    <body>
     <div id="pt-pagelet-content-1" class="pagelet-container" style="display:
inline;">
      <span xmlns:pt="http://www.plumtree.com/xmlschemas/ptui/">
      Pagelet links:
      <br/>
      <a href="http://proxy:port/inject/RP_PID393219_I1/headpagelet1.html">The
first pagelet</a>
      <br/>
      <a href="http://proxy:port/inject/RP_PID393219_I1/headpagelet2.html">The
second pagelet</a>
      <br/>
      <a href="http://proxy:port/inject/RP_PID393219_I1/csapipagelet.html">The
csapi pagelet</a>
      <br/>
      <a href="http://proxy:port/inject/RP_PID393219_I1/linkspagelet.html">This
pagelet</a>
      <br/>
      </span>
     </div>
    </body>
   </html>
  </iframe>
 </body>
</html>
```

63.2.2.2.2 Data Retrieval APIs Two REST APIs are available to retrieve data from the Pagelet Producer:

- **Pagelet API**: Allows remote applications to retrieve pagelet data from the Pagelet Producer. (See Example 63–1, "All Pagelets" and Example 63–2, "Pagelets By Library and Name".)
- **Resource API**: Allows remote applications to retrieve resource data from the Pagelet Producer. (See Example 63–3, "All Resources" and Example 63–4, "Resource By Name".)

The base URL for all requests is `http://`*`host:port`*`/pagelets/restservice/pageletproducer/`.

Example 63–1 All Pagelets

```
http://host:port/pagelets/restservice/pageletproducer/pagelets/
http://host:port/pagelets/restservice/pageletproducer/pagelets/?format=xml
```

Example 63–2 Pagelets By Library and Name

```
http://host:port/pagelets/restservice/pageletproducer/pagelet/libraryname/pageletn
ame/
http://host:port/pagelets/restservice/pageletproducer/pagelet/libraryname/pageletn
ame/?format=xml
```

Example 63–3 All Resources

```
http://host:port/pagelets/restservice/pageletproducer/resources/
http://host:port/pagelets/restservice/pageletproducer/resources/
```

Example 63–4 Resource By Name

```
http://host:port/pagelets/restservice/pageletproducer/resource/name
http://host:port/pagelets/restservice/pageletproducer/resource/name/?format=xml
```

63.2.2.3 Using Automatic Resizing with IFrames

The Pagelet Producer pagelet inject API can automatically resize the IFrame that encapsulates pagelet content. The resizing is done so that the IFrame stretches to fit the content within. To use this feature, the ifwidth and ifheight parameters must be set to 'auto' as shown in the example below:

```
http://proxy:port/inject/v2/pagelet/samples/linkspagelet?content-type=iframe&csapi
=true&ifheight=auto&ifwidth=auto&ifclass=myclass
```

In addition, this feature relies on an external page on the same domain as the consumer page. This page is included into the pagelet IFrame as an internal hidden IFrame. This page collects the sizing information and passes it on to the parent consumer page. This page must be deployed in the same directory as the consumer page.

The example below resizes the pagelet IFrame after it finishes loading. To add dynamic auto-resizing capabilities to user interaction activities after the initial load, simply add more event listeners for mouse and keyboard events.

```
<html>
  <head>
    <title>Resizing Page</title>
    <script type="text/javascript">
  function onLoad() {
    var params = window.location.search.substring( 1 ).split( '&' );
    var height;
    var width;
    var iframe;

    for( var i = 0, l = params.length; i < l; ++i ) {
      var parts = params[i].split( '=' );
      switch( parts[0] ) {
        case 'height':
          height = parseInt( parts[1] );
          break;
        case 'width':
          width = parseInt( parts[1] );
          break;
        case 'iframe':
```

```
          iframe = parts[1];
          break;
       }
     }
   window.top.updateIFrame( iframe, height, width );
   }

   if (window.addEventListener) {
     window.addEventListener("load", onLoad, false)
   } else if (window.attachEvent) {
      window.detachEvent("onload", onLoad)
      window.attachEvent("onload", onLoad)
   } else {
     window.onload=onLoad
   }
   </script>
 </head>
<body>
</body>
</html>
```

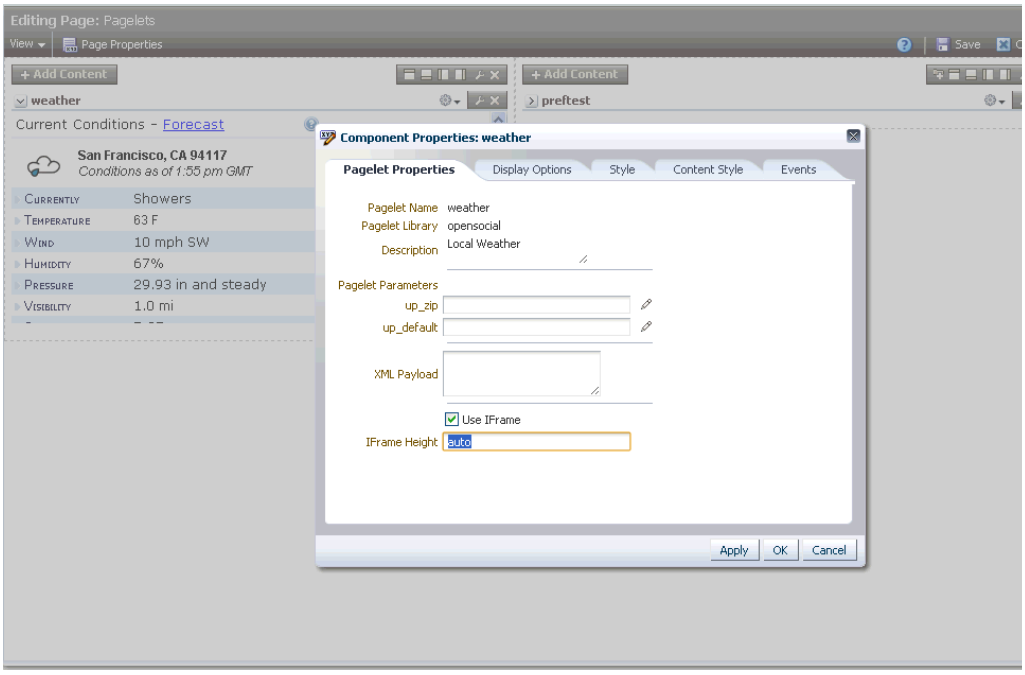
63.2.3 Adding a Pagelet to a Page in WebCenter Portal: Spaces

In WebCenter Portal: Spaces, you can add a pagelet to a page using Composer. By default, pagelets appear in the Mash-Ups folder in the Resource Catalog. To add a pagelet to a page, navigate to the pagelet in the Resource Catalog and select it.

For detailed information about how to add resources to pages in Spaces, see the section "Adding Resource Catalog Components to Pages" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

To configure a pagelet within a page, view the page in Edit mode and click the edit button (wrench icon) for the pagelet. The Pagelet Properties tab in the Component Properties dialog allows you to define pagelet parameters and IFrame options.

Figure 63–4 Component Properties Dialog: Pagelet Properties



63.3 Building Pagelets Using the Pagelet Producer

This section provides information on building pagelets using the Pagelet Producer.

- Section 63.3.1, "Using the Adaptive Pagelet Scripting Framework"
- Section 63.3.2, "Using Pagelet Producer Credential Mapping"
- Section 63.3.4, "Debugging Pagelets"

63.3.1 Using the Adaptive Pagelet Scripting Framework

The **Adaptive Pagelet Scripting Framework** is a client-side JavaScript library that provides services to CSP pagelets and proxied pages. This section explains how to use the scripting framework to implement dynamic functionality in pagelets.

- Section 63.3.1.1, "Handling Structured HTTP Responses": The adaptive pagelet scripting framework can be used as a client-side response handler for structured HTTP, typically encoded as XML.
- Section 63.3.1.2, "Using Event Notification": A pagelet can respond when specific events happen, such as when the page loads or when the browser focus changes.
- Section 63.3.1.3, "Using In-Place Refresh": Pagelets can reload their internal content without refreshing the page.
- Section 63.3.1.4, "Using Session Preferences": Browser-level variables can be stored and shared among pagelets, even if they are not on the same page. For example, a value entered by the user in one pagelet can be retrieved by another. The scripting framework acts as an intermediary, allowing all pagelets access to all values stored in a common session.

For a full list of classes and methods, see the JSPortlet API documentation.

For additional information on adaptive pagelets, see Section 63.3.1.5, "Adaptive Pagelet Development Tips".

63.3.1.1 Handling Structured HTTP Responses

In many cases it can be expensive and inefficient to send large amounts of HTML back in response to some HTTP request, if only a small part of the user interface needs to be changed. This is especially true with more complex user interfaces. In these cases, the response can be encoded in XML. The client-side response handler can then parse the XML, and update the user interface (or perform some other action) based on that response. Use the Structured Response design pattern to redraw a small area of the user interface after making an HTTP request, or to access a simple HTTP/URI type web service from a pagelet. The example code below (structuredresponse_portlet.html) accesses an RSS feed from a selection of news sites.

```
<!-- jsxml includes -->
<a id="imgServerHref" href="/images/plumtree" style="display:none"></a>
<script type="text/javascript"
src="/images/plumtree/common/private/js/PTLoader.js"></script>
<script type="text/javascript">
var oImgServer = new Object();
oImgServer.location = document.getElementById('imgServerHref').href;
var imageServerURL = document.getElementById('imgServerHref').href;
var imageServerConnectionURL = oImgServer.location;
new PTLoader(imageServerURL, imageServerConnectionURL).include('jsxml','en');
</script>

<!-- jscontrols includes -->
<link rel="stylesheet" type="text/css"
href="/portal-remote-server/js/jscontrols/styles/css/PTMenu.css"/>
<link rel="stylesheet" type="text/css"
href="/portal-remote-server/js/jscontrols/styles/css/PTRichTextEditor.css"/>
<script type="text/javascript"
src="/portal-remote-server/js/jscontrols/strings/PTControls-en.js"></script>
<script type="text/javascript"
src="/portal-remote-server/js/jscontrols/PTControls.js"></script>

<!-- Inline JS helper functions -->
-->

<script defer type="text/javascript" id="structured-response-portlet-A-script">
// Function that gets the RSS XML feed found at the specified url
getRSSFeed = function(url)
  {
  // First clear out any existing rows in the table
  channelTable.clearRows();

  // Force the transformer to fix up the url
  var oURL = new Object();
  oURL.location = url;

  // Do the http get
  var get = new PTHTTPGETRequest(oURL.location, handleRSSResponse);
  get.invoke();
  }

// Function that handles the RSS XML response and updates the table based on the
RSS items
handleRSSResponse = function(response)
  {
```

```
  // Get the rss xml
  var xml = response.responseText;
  if (!xml || xml.indexOf('<?xml') == -1) { return; }

  // Parse into a dom, and get the channel node
  var xmlDOM = new PTXMLParser(xml);
  var rssNode = xmlDOM.selectSingleNode('rss');
  var channelNode = rssNode.selectSingleNode('channel');

  // Get the channel title and set the status bar text in the table
  var channelTitle = channelNode.selectSingleNode('title').getNodeValue();
  channelTable.statusBarText = '<b>Loaded Channel</b>: ' + channelTitle;

  // Get channel item nodes
  var itemNodes = channelNode.selectNodes('item');

  // Build table rows
  channelTable.rows = new Array();
  for (var i=0; i<itemNodes.length; i++)
    {
    var itemNode = itemNodes[i];

    // Get channel item properties
    var itemTitle = itemNode.selectSingleNode('title').getNodeValue();
    var itemLink = itemNode.selectSingleNode('link').getNodeValue();
    var itemDescription = itemNode.selectSingleNode('description').getNodeValue();
    if (itemNode.selectSingleNode('author'))
      var itemAuthor = itemNode.selectSingleNode('author').getNodeValue();
    if (itemNode.selectSingleNode('category'))
      var itemCategory = itemNode.selectSingleNode('category').getNodeValue();
    if (itemNode.selectSingleNode('pubDate'))
      var itemPubDate = itemNode.selectSingleNode('pubDate').getNodeValue();

    // Create a row and add it to the table
    var row = new PTRow();
    row.parent = channelTable;
    row.id = i;
    row.uid = i;
    row.previewText = itemDescription;
    row.link = itemLink;
    row.columnValues[0] = new PTTextColumnValue(itemTitle);
    row.columnValues[1] = new PTTextColumnValue(itemCategory);
    row.columnValues[2] = new PTTextColumnValue(itemAuthor);
    row.columnValues[3] = new PTTextColumnValue(itemPubDate);
    channelTable.rows[channelTable.rows.length] = row;
    }

  // Redraw the table
  channelTable.draw();
  }
</script>

<b>Select RSS Feed:</b>
<a href="#"
onclick="getRSSFeed('http://www.wired.com/news/feeds/rss2/0,2610,,00.xml'); return
false;">Wired News</a>
<a href="#" onclick="getRSSFeed('http://news.com.com/2547-1_3-0-5.xml'); return
false;">CNET News.com</a>
<a href="#"
onclick="getRSSFeed('http://partners.userland.com/nytRss/nytHomepage.xml'); return
```

```
false;">NY Times</a>
<br><br>

<!-- Set up a table control to display channel items -->
<div id="channelTableContainer"></div>
<script defer type="text/javascript">
  var channelTable = new PTTableControl();
  channelTable.locale = 'en_US';
  channelTable.objName = 'channelTable';
  channelTable.container = 'channelTableContainer';
  channelTable.baseURL =
'/imageserver/plumtree/common/private/portal-remote-server/js/jscontrols/1/';
  channelTable.statusBarText = 'No RSS Feed Selected';
  channelTable.rowDetailAction = new
PTJavaScriptAction('window.open(\'${ROW.link}\');');
  channelTable.columns[0] = new PTColumn();
  channelTable.columns[0].name = 'Title';
  channelTable.columns[0].width = '40%';
  channelTable.columns[1] = new PTColumn();
  channelTable.columns[1].name = 'Category';
  channelTable.columns[1].width = '20%';
  channelTable.columns[2] = new PTColumn();
  channelTable.columns[2].name = 'Author';
  channelTable.columns[2].width = '20%';
  channelTable.columns[3] = new PTColumn();
  channelTable.columns[3].name = 'Publication Date';
  channelTable.columns[3].width = '20%';
  channelTable.areColumnsResizable = true;
  channelTable.clientSortEnabled = true;
  channelTable.scrollHeight = 250;
  channelTable.init();
  channelTable.draw();
</script>
</div>
```

63.3.1.2 Using Event Notification

The adaptive pagelet scripting framework allows pagelets to respond to both page-level events and custom events raised by other pagelets.

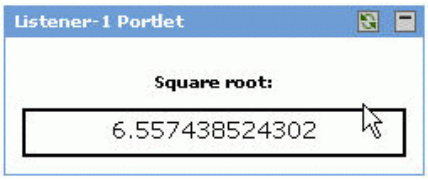
The `registerForWindowEvent` and `registerOnceForWindowEvent` methods in the scripting framework provide pagelets with access to page-level events. For a complete list, see Section 63.3.1.2.1, "Page-Level Events for Use with the Scripting Framework". To register for notification of these events, pass in the name of the event and the name of the method that should be called when it occurs. When a page-level event is raised, the JavaScript event object is passed to the event handler as an argument. The scripting framework also allows pagelets to raise and respond to custom events using `raiseEvent` and `registerForEvent`. The Broadcast-Listener design pattern illustrates an important example of using notification services with session preferences. Users can select an item or perform some other action in a "broadcast" pagelet, which causes the content in other related "listener" pagelets to be redrawn. In the following example, the broadcast pagelet displays a form that allows you to enter a number in a text box.

Figure 63–5 Broadcast Portlet



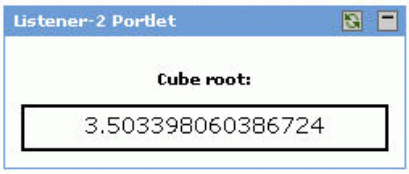
When the user enters a number in the text box, the values in the listener pagelets change. The first listener pagelet displays the square root of the number entered in the broadcast pagelet.

Figure 63–6 Listener -1 Portlet



The second listener pagelet displays the cube root of the number entered in the broadcast pagelet.

Figure 63–7 Listener -2 Portlet



The following steps summarize how the pagelets work:

- On load, each listener pagelet calls its own instance method (`registerForEvent`) to register for events of type 'onBroadcastUpdate'.
- On each onkeyup event that occurs in the "Enter number" text box, the broadcast pagelet sets a session preference to the value entered in the text box, and calls its own instance method (`raiseEvent`) to raise an event of type 'onBroadcastUpdate'.
- When the onBroadcastUpdate event is raised or the page is reloaded, each listener pagelet retrieves the session preference set by the broadcast pagelet and computes a new value to display based on the value of the preference.

Broadcast Pagelet

```
<div style="padding:10px;" align="center">
<p><b>Enter number:</b>
 <input type="text"
style="font-size:22px;font-weight:bold;text-align:center;"
id="broadcast_prefName" value="4" size="7" onkeyup="broadcast_
setPrefs(this.value)"></p>
<br>
</div>
```

```
<script type="text/javascript">
function broadcast_setPrefs(val)
{
    var prefName = 'broadcastNumber';
    var prefValue = val;
    PTPortlet.setSessionPref(prefName,prefValue);
    var broadcastPortlet =
PTPortlet.getPortletByGUID('{D9DFF3F4-EAE7-5478-0F4C-2DBD94444000}');

      if (!broadcastPortlet)
    {
        broadcast_debug('Could not locate PTPortlet object which corresponds to
<b>Broadcast Portlet</b> on page.');
        return;
        }
      broadcast_debug('<b>Broadcast Portlet</b> raising onBroadcastUpdate
event.');
    broadcastPortlet.raiseEvent('onBroadcastUpdate',false);
}
function broadcast_debug(str)
{
    if (window.PTDebugUtil)
    {
        PTDebugUtil.debug(str);
    }
}
</script>
```

Listener Pagelet #1

```
<div style="padding:10px;" align="center">
<p><b>Square root:</b>
<div style="height:21px;border:2px solid
black;padding:2px;overflow:visible;font-size:14px;"id="listener1-swatch">
</div>
</div>

<script>
function listener1_update()
{
    var broadcastNumber = parseFloat(PTPortlet.getSessionPref('broadcastNumber'));
    if (isNaN(broadcastNumber))
    {
        listener1_error('<b>Listener-1 Portlet</b> cannot parse number from
session pref broadcastNumber');
        return;
    }

      listener1_debug('<b>Listener-1 Portlet</b> computing square root of ' +
broadcastNumber);
      var swatch = document.getElementById('listener1-swatch');
    swatch.innerHTML = Math.sqrt(broadcastNumber);
}

function listener1_debug(str)
{
    if (window.PTDebugUtil)
    {
        PTDebugUtil.debug(str);
    }
```

```
}

function listener1_error(str)
{
    if (window.PTDebugUtil)
    {
        PTDebugUtil.error(str);
    }
}

function listener1_getPortlet()
{
    var portletGUID = '{D9DFF3F4-EAE7-5478-0F4C-2DBDB4F4A000}';
    var listener1Portlet = PTPortlet.getPortletByGUID(portletGUID);
    return listener1Portlet;
}

var listener1Portlet = listener1_getPortlet();
if (listener1Portlet)
{
    listener1Portlet.registerForEvent('onBroadcastUpdate','listener1_update');
    listener1_debug('<b>Listener-1 Portlet</b> registered refreshOnEvent for event
onBroadcastUpdate');
    listener1Portlet.registerForEvent('onload','listener1_update');
}

</script>
```

Listener Pagelet #2

```
<div style="padding:10px;" align="center">
<p><b>Cube root:</b>
<div style="height:21px;border:2px solid
black;padding:2px;overflow:visible;font-size:14px;"id="listener2-swatch">
</div>
</div>

<script>
var listener2_oneThird = (1/3);
function listener2_update()
{
    var broadcastNumber = parseFloat(PTPortlet.getSessionPref('broadcastNumber'));
    if (isNaN(broadcastNumber))
    {
        listener2_error('<b>Listener-2 Portlet</b> cannot parse number from
session pref broadcastNumber');
        return;
    }

    listener2_debug('<b>Listener-2 Portlet</b> computing square root of ' +
broadcastNumber);
    var swatch = document.getElementById('listener2-swatch');
    swatch.innerHTML = Math.pow(broadcastNumber,listener2_oneThird);
}

function listener2_debug(str)
{
    if (window.PTDebugUtil)
    {
        PTDebugUtil.debug(str);
    }
```

```
}

function listener2_error(str)
{
    if (window.PTDebugUtil)
    {
        PTDebugUtil.error(str);
    }
}

function listener2_getPortlet()
{
    var portletGUID = '{D9DFF3F4-EAE7-5478-0F4C-2DBDCA1C7000}';
    var listener2Portlet = PTPortlet.getPortletByGUID(portletGUID);
    return listener2Portlet;
}

var listener2Portlet = listener2_getPortlet();
if (listener2Portlet)
{
    listener2Portlet.registerForEvent('onBroadcastUpdate','listener2_update');
    listener2_debug('<b>Listener-2 Portlet</b> registered refreshOnEvent for event
onBroadcastUpdate');
    listener2Portlet.registerForEvent('onload','listener2_update');
}

</script>
```

63.3.1.2.1 Page-Level Events for Use with the Scripting Framework The scripting framework automatically has access to the following page-level events.

Table 63–2 Page-Level Events

Event	Triggered:
onload	immediately after the browser loads the page
onbeforeunload	prior to a page being unloaded (browser window closes or navigates to different location)
onunload	immediately before the page is unloaded (browser window closes or navigates to different location)
onactivate	the page is set as the active element (receives focus)
onbeforeactivate	immediately before the page is set as the active element (receives focus)
ondeactivate	when the active element is changed from the current page to another page in the parent document
onfocus	when the page receives focus
onblur	when the page loses focus
oncontrolselect	when the user is about to make a control selection of the page
onresize	when the size of the page is about to change
onresizestart	when the user begins to change the dimensions of the page in a control selection
onresizeend	when the user finishes changing the dimensions of the page in a control selection
onhelp	when the user presses the F1 key while the browser is the active window

Table 63–2 (Cont.) Page-Level Events

Event	Triggered:
onerror	when an error occurs during page loading
onafterprint	immediately after an associated document prints or previews for printing

63.3.1.3 Using In-Place Refresh

To refresh pagelet content in place, without affecting other content on the page, use the scripting framework to implement in-place refresh.

Many pagelets display data that is time sensitive. In some cases, users should be able to navigate across links within a pagelet without changing or refreshing the rest of the page. You can refresh pagelet content on command, associate the refresh action with an event (`refreshOnEvent`), or program the pagelet to refresh at a set interval (`setRefreshInterval`). The scripting framework also contains methods for expanding and collapsing pagelets. In the simplified example below, the refresh pagelet displays a "Refresh Portlet" button. Clicking the button updates the date and time displayed in the pagelet.

Figure 63–8 Refresh Portlet



The in-place refresh is executed by calling the `refresh()` method on the pagelet object instance. You can also set a new URL to be displayed within the pagelet upon refresh. (The title bar cannot be altered on refresh.)

```
<div style="padding:10px;" align="center">
<p><button onclick="refresh_portlet()">Refresh Portlet</button></p>
<p><b>Current time is:</b><br> <span id="refreshTimeSpan"></span></p>
</div>

<script type="text/javascript">
function refresh_portlet()
{
var refreshPortlet = PTPortlet.getPortletByID($PORTLET_ID$);
if (!refreshPortlet)
  {
  refresh_debug('Could not locate PTPortlet object which corresponds to <b>Refresh
Portlet</b> on page.');
  return;
  }
refresh_debug('<b>Refresh Portlet</b> calling refresh() method.');
refreshPortlet.refresh();
}

function refresh_debug(str)
{
if (window.PTDebugUtil)
  {
```

```
  PTDebugUtil.debug(str);
  }
}

var t = new Date();
document.getElementById('refreshTimeSpan').innerHTML = t;
</script>
```

63.3.1.4 Using Session Preferences

To store and share settings within the client browser, use session preferences.

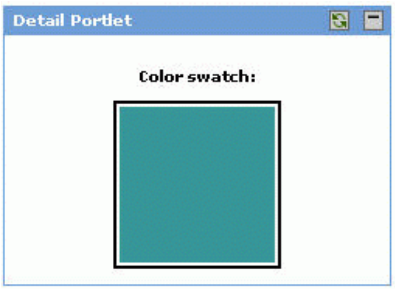
Pagelets can use preferences to communicate with each other, but accessing preferences usually requires a round trip to a database. Session preferences provide a way to store and share settings in the user's session within the client browser. The Master-Detail design pattern illustrates the most basic usage of session preferences. This design pattern splits control and display between two pagelets. For example, the "master" pagelet could summarize data in list form, and the "detail" pagelet could display details on each data item in response to user selection. In the example below, the master pagelet displays a form that allows you to enter a color code in a text box.

Figure 63–9 Color Code



When the user enters a color code in the text box, the color in the detail pagelet changes.

Figure 63–10 Color Swatch



For each `onkeyup` event that occurs in the "Enter color" text box in the master pagelet, the following steps are executed:

1. The master pagelet sets the session preference using the current value of the text box.
2. The master pagelet calls an update method on the detail pagelet.
3. The detail pagelet retrieves the session preference to get the color value.
4. The detail pagelet redraws its color swatch area to reflect the new color value.

> **Note:** Shared session preferences must be specified by name on the Preferences page for the pagelet in the Pagelet Producer Console or they will not be sent to the pagelet.

The adaptive pagelet scripting framework provides an easy way to detach the relationship between pagelets and use a common event interface for communication.

> **Note:** The example below is oversimplified; the master pagelet makes a direct call to a JavaScript method of the detail pagelet. Unless the master pagelet takes extra measures to ensure that the detail pagelet is actually present on the same page, calls from master to detail could generate errors.

Master Pagelet

```
<div style="padding:10px;" align="center">
<p><b>Enter color:</b>  
<input type="text" style="font-size:22px;font-weight:bold;text-align:center;"
id="master_prefName"
value="#FFFFFF" size="8" onkeyup="master_setPrefs(this.value)"></p><br>
</div>

<script type="text/javascript">
function master_setPrefs(val)
{
var prefName = 'masterColor';
var prefValue = val;
PTPortlet.setSessionPref(prefName,prefValue);

master_debug('<b>Master Portlet</b> called
PTPortlet.setSessionPref(\'masterColor\',\'' + prefValue + '\').');

if (window.detail_update)
  {
  master_debug('<b>Master Portlet</b> calling detail_update().');
  detail_update();
  }
else
  {
  master_debug('Could not locate portlet <b>Detail Portlet</b> on page.');
  }
}
function master_debug(str)
{
if (window.PTDebugUtil)
  {
  PTDebugUtil.debug(str);
  }
}
</script>
```

Detail Pagelet

```
<div style="padding:10px;" align="center">
<p><b>Color swatch</b>  
<div style="width:100px;height:100px;border:2px solid
black;padding:2px;"id="detail-swatch"></div>
```

```
<script>
function detail_update()
{
var color = PTPortlet.getSessionPref('masterColor');
detail_debug('<b>Detail Portlet</b> received value="' + color + '" for
PTPortlet.getSessionPref(\'masterColor\')');

var swatch = document.getElementById('detail-swatch');
if (swatch)
  {
  swatch.innerHTML = '<div style="background-color:' + color +
';width:100%;height:100%;"></div>';
  }
else
  {
  detail_debug('<b>Detail Portlet</b> cannot find \'detail-swatch\' DIV
element.');
  }
}

function detail_debug(str)
{
if (window.PTDebugUtil)
  {
  PTDebugUtil.debug(str);
  }
}
</script>
```

63.3.1.5 Adaptive Pagelet Development Tips

These tips apply to most pagelets that use the adaptive pagelet scripting framework.

- **Use unique names for all forms and functions.** Use the GUID of a pagelet to form unique names and values to avoid name collisions with other code on the page.
- **Proxy all URLs.** You cannot make a request to a URL with a host/port that is different from that of the calling page. All URLs requested through JavaScript must be proxied. For details, see Section 63.3, "Building Pagelets Using the Pagelet Producer"
- **If you are using the adaptive pagelet scripting framework, check for support.** It is good practice to include code that determines whether or not the component is present. Ideally, your pagelet should be able to handle either situation. The simplest solution is to precede your code with an If statement that alerts the user if the scripting framework is not supported.

  ```
  <script>
  if (PTPortlet == null)
    {
    if (document.PCC == null)
      {
      alert("This pagelet only works in portals that support the JSPortlet API .
  The pagelet will be displayed with severely reduced
      functionality. Contact your Administrator.");
      }
    }
  else
    {
    [scripting code here]
    }
  ```

```
</script>
```

- **Close all popup windows opened by a pagelet when the window closes.** The scripting framework can be used to close popup windows using the `onunload` event.
- **Add all required JavaScript to the page in advance.** Browsers might not process script blocks/includes added to the page through the `innerHTML` property.
  - Microsoft Internet Explorer: Add the defer attribute to the script tag.
  - Netscape: Use RegExp to parse the response and look for the script, then eval it.
- **JavaScript HTTP and proxied HTTP must use the same authentication credentials.** JavaScript brokered HTTP requests send the same authentication token (cookie) as when you make a normal gatewayed HTTP request.

63.3.2 Using Pagelet Producer Credential Mapping

The Pagelet Producer manages authentication with each proxied application based on the settings defined for the associated resource.

The Pagelet Producer provides an API for creating custom mappings to external credential stores, allowing you to authenticate users against a custom credential source.

The `IVendorCredentialMapper` interface defines the Pagelet Producer interface for objects capable of obtaining an appropriate set of credentials needed for secondary authentication for a particular user in an application. To implement this interface, follow the directions below.

1. Create a java class that implements the `com.plumtree.runner.credentialmapper.IVendorCredentialMapper` interface.
2. Map the `getCredential` and `setCredential` methods of this interface to your credential vault. The simplified example below uses an internal class called `VConnector` and calls `VConnector.getInstance().getCredentialsForDomain`. Note: This step is vendor-specific. It will probably include a network hop, since the credential store will most likely reside on another server. You must give the mapper a unique name, and localized ones if necessary. See the IVendorCredentialMapper API documentation for all required names.
3. Compile the class into a jar. The build process must link to `common.jar`, included with the Pagelet Producer distribution.
4. Load the custom vault into the Pagelet Producer. For details, see the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
5. Restart the Pagelet Producer server (both proxy and adminui if they are on separate servers). The custom credential vault should show up in the list of credential sources on the Credential Mapping page of the Resource editor.

The example below is simplified for illustration purposes.

```
package com.oracle.credentialvault;
import com.oracle.connector.CredentialsSet;
import com.oracle.connector.VConnector;
import com.plumtree.runner.credentialmapper.Credential;
import com.plumtree.runner.credentialmapper.IVendorCredentialMapper;
```

```
public class OracleCredentialVault implements IVendorCredentialMapper {

  /*
  * Pagelet Producer will pass credential types as following:
  * Runner_*, where * is what the credential value type associated with this login
  * form in the adminui. For example, if the credential value type is
  *  'username' then "Runner_username" will be passed to the mapper.
  */
  public Credential getCredential(String initiator, String credType)
  {
     System.out.println("OracleCredentialVault::getCredential, initiator: " +
initiator + ", credType: " + credType);

    /*
     * Since this vault stores credentials per user and domain, we need to devise
     * a scheme to map Pagelet Producer's credential type to a domain.
     * One way to do this is to specify the credential type as something like:
     * "domain_type", which would translate to credTypes like:
     * Runner_domain.com_username and Runner_domain.com_password
     */

    String username = initiator.toLowerCase(); // assume that the vault stores all
usernames in lowercase
    String domain = "oracle.com"; //getDomain(credType);
    String type = credType; //getType(credType);

    CredentialsSet credSet =
VConnector.getInstance().getCrededentialsForDomain(username, domain);
    if( credSet != null ) {
        System.out.println("OracleCredentialVault::getCredential, found vault set:
" + credSet.toString() + ", returning type = " + type);
        return new Credential(credSet.getCredential(type));
    } else {
       System.out.println("OracleCredentialVault::getCredential, found null vault
set");
       return null;
    }
  }

  public String getDescription(String userLocale)   {
    return "Test mapper that mimics a mapper between Pagelet Producer and a
credential vault that associates credentials with a username/domain relationship";
  }

  public String getName() {
     return "OracleCredentialVault";  }

  public String getName(String userLocale) {
     return "OracleCredentialVault"; }

  public String getVendorName(String userLocale) {
     return "Oracle"; }

  public boolean setCredential(String initiator, Credential credential, String
credType)
  {
     System.out.println("OracleCredentialVault::setCredential, initiator: " +
initiator + ", credType: " + credType + ", Credential: " +
credential.getCredentialValue());
```

```
     String username = initiator.toLowerCase(); // assume that the vault stores
all usernames in lowercase
     String domain = "oracle.com"; //getDomain(credType);
     String type = credType; //getType(credType);

     System.out.println("OracleCredentialVault::setCredential setting username: "
+ credential.getCredentialValue());
     CredentialsSet userCredSet =
VConnector.getInstance().getCrededentialsForDomain(username, domain);
     userCredSet.setCrededential(type, credential.getCredentialValue());
     VConnector.getInstance().setCrededentialsForDomain(username, domain,
userCredSet);
     return true;
  }

  public boolean supportsCredentialsEditing()
  {
        // We can set new credentials using this vault
        return true;
    }
  /*
  private String getDomain(String credType)
  {
     int dstart = credType.indexOf("_");
     int dend = credType.indexOf("_", dstart+1);
     String domain = credType.substring(dstart+1, dend);
     System.out.println("TestMapper::getDomain, reading domain as: " + domain);
     return domain;
  }
  */
  /*
  private String getType(String credType)
  {
     int dstart = credType.indexOf("_");
     dstart = credType.indexOf("_", dstart+1);
     String type = credType.substring(dstart+1, credType.length());
     System.out.println("TestMapper::getType, reading type as: " + type);
     return type;
  }
  */
  /*
  private String doGetPropertyValue(String principal, String property)
  {
     return doGetPropertyValue(principal, property, ",", "=");
  }
  */
  /*
  private String doGetPropertyValue(String principal, String property, String
propDelim, String valueDelim)
  {
     int propertyindex = principal.toLowerCase().indexOf(property.toLowerCase());
     String uname = null;
     if( propertyindex != -1)
     {
        // found a property occurence
        int beginIndex = propertyindex;
        int endIndex = principal.toLowerCase().indexOf(propDelim.toLowerCase(),
beginIndex);
        String prop = null;
        if(endIndex != -1)
```

```
        {
           prop = principal.subSequence(beginIndex, endIndex).toString().trim();
         } else {
           prop = principal.subSequence(beginIndex,
principal.length()).toString().trim();
         }
         if( prop != null )
         {
           int valueIndex = prop.toLowerCase().indexOf(valueDelim);
           if(valueIndex != -1)
           {
              uname = prop.subSequence(valueIndex + valueDelim.length(),
prop.length()).toString().trim();
           }
         }
     }
     return uname;
  }
   */
}
```

For details on configuring resources to use credential mappings, see the section "Creating Pagelet Producer Resources and Pagelets" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

63.3.3 Modifying Pagelet Functionality at Runtime

The Pagelet Producer allows you to modify pagelet functionality at runtime using custom injectors and parsers.

63.3.3.1 Creating Web Injectors

A web injector inserts content into a specified location in the proxied resource page. The content may be any text, including HTML, CSS, JavaScript, and pagelet declarations. An empty injector may also be used to remove unwanted content from a page. Injectors cannot be created for OpenSocial resources. While injecting simple HTML content has a limited use case, you can also inject JavaScript that directly modifies the pagelet's HTML markup.

To create a web injector, select the **Injectors** section under the resource you want to use and click the **Create** icon in the toolbar.

The injector can be applied to a subset of the resource by typing a URL pattern into the **URL Filter** field. The injector will be applied only to those URLs within the resource that begin with the text in the URL Filter box. If the box is empty or contains only a '/', the injector will be applied to the entire resource.

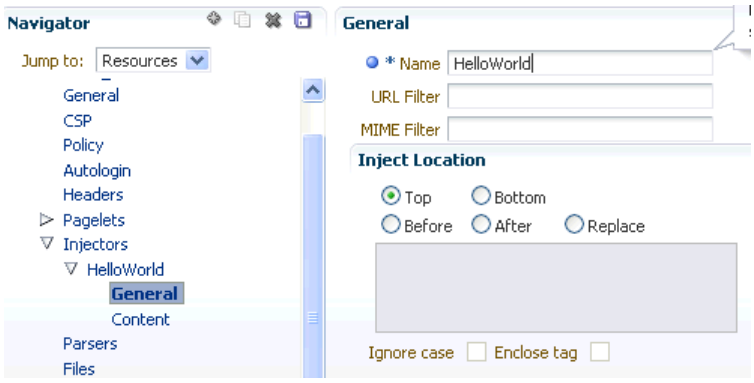
To restrict the injector to specific kinds of content, type a comma separated list of MIME types in the **MIME Filter** box. For example, *text/html* restricts the injector to HTML content, while *text/css* only restricts the injector to CSS content.

Define where in the resource's output the injection will be made:

- **Top** puts the content first in the page. Do not use this option to inject pagelet declarations.
- **Bottom** puts the content last in the page.
- The **Before/After/Replace** options put the content into the page relative to the unique string specified in the field provided. You can choose to ignore the case of the string by selecting **Ignore case**.

The **Enclose tag** option identifies the unique string and replaces both the text and the enclosing tag with the content specified on the next page.

Figure 63–11 Pagelet Producer Console: Injectors - General Page



Enter the content to be injected on the **Content** page. This content may be any text, including HTML, CSS, JavaScript, and pagelet declarations.

For example, the following code could be injected at the top of a page. This example registers a handler function with the page load event and then uses the handler to modify the page markup (by finding and hiding the header and footer).

```
<script type="text/javascript">
if (window.addEventListener) {
  window.addEventListener('load', hideHeaderFooter, false);
} else if (document.attachEvent) {
  window.attachEvent('onload', hideHeaderFooter);
}
function hideHeaderFooter() {
  var header = null;
  // find the header table by class
  if (document.getElementsByClassName) {
    header = document.getElementsByClassName('page_header')[0];
  } else {
    // for older versions of IE
    var tables = document.getElementsByTagName('table');
    for (var table in tables) {
      if (table.class == 'page_header') {
        header = table;
        break;
      }
    }
  }
  if (header != null) {
    header.style.display = 'none';
  }
  // now find and hide the footer (easier to find, since it has an id)
  var footer = document.getElementById('t23PageFooter');
  if (footer != null) {
    footer.style.display = 'none';
  }
}
</script>
```

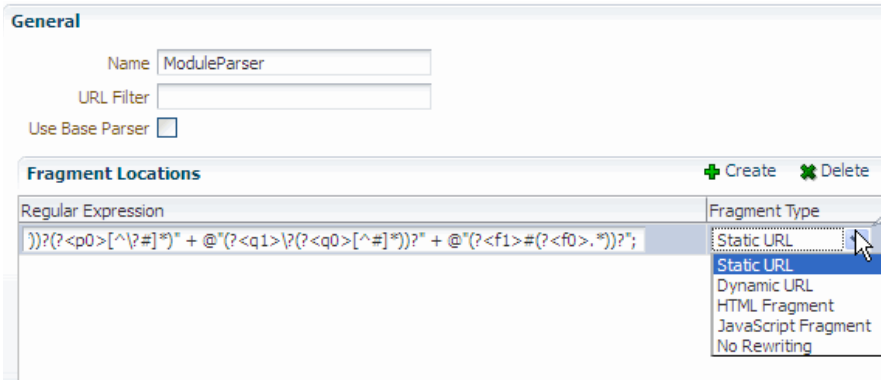
63.3.3.2 Creating Custom Parsers

Custom parsers allow you to supplement or change built-in logic for parsing content and finding URLs. When the built-in parsers fail to identify URLs or identify sections that must not be rewritten as URLs, custom parsers can be used to change the default behavior. Parsers cannot be created for WSRP or Oracle JPDK portlet producers or for OpenSocial gadget producers.

To create a custom parser, select the **Parsers** section under the resource you want to use and click the **Create** icon in the toolbar.

- Enter a **Name** for the parser.
- The parser can be applied to a subset of the resource by typing a URL pattern into the **URL Filter** box. The parser will be applied only to those URLs within the resource that begin with the text in the URL Filter box. If the box is empty or contains only a '/', the parser will be applied to the entire resource.
- To add a new parsing rule, click **Create** to add a new row to the **Fragment Locations** section.
- In the **Regular Expressions** column, enter the regular expression for identifying the URL fragment that should be transformed. The first grouping expression (in parentheses) identifies the fragment, and the rest of the expression provides the context for finding it.
- Choose the **Fragment Type** to define how the selected location should be parsed:
  - **Static URLs** are transformed on the server.
  - **Dynamic URLs** are transformed using JavaScript on the client.
  - **HTML Fragment** and **Javascript Fragment** types are used for content that is embedded in another content type, such as XML.
  - **No Rewriting** specifies a location that should not be searched for URLs. This option is used to prevent rewriting of markup mistakenly identified as URLs.
- Click the **Save** icon in the navigator toolbar.

Figure 63–12 Pagelet Producer Console: Parsers - General Page

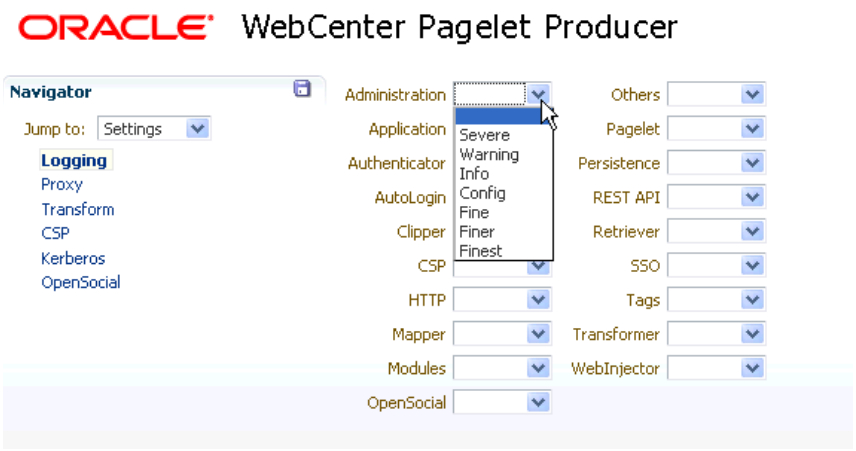


For example, the regular expression `XMLFile=(.*?)"` would identify URLs to XML files defined within a tag, as in `<embed src="/i/flashchart/anychart_5/swf/OracleAnyChart.swf?>XMLFile=http://apex.oracle.com/pls/apex/apex_util.flash?p=53557:1:74175370346201:FLOW_FLASH_CHART5_R45192463162785599619_en"`.

63.3.4 Debugging Pagelets

The Pagelet Producer provides advanced logging traces for debugging pagelets. Logging is configured in the Settings section of the Pagelet Producer Console, where you can define different levels of logging for each Pagelet Producer component. (For details on Pagelet Producer configuration settings, see *Managing the Oracle WebCenter Portal's Pagelet Producer* in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.)

Figure 63–13 Pagelet Producer Console: Settings - Logging



The Pagelet Producer logs messages to the standard Oracle Diagnostic Logging facility. In Oracle WebLogic Server, the logs are stored at `user-projects/domains/<domain>/servers/<server>/logs/<server>-diagnostic.log`.

63.3.4.1 HTTP Tracing

To view the HTTP requests and responses received and sent by the Pagelet Producer, set the **HTTP** component on the Logging Settings page to **Finest**. The example traces below were captured from a test environment.

Example 63–5 HTTP Request Received By Pagelet Producer

```
URL: http://10.148.118.211:7001/pagelets/bidwiki/includes/js/ajax.js
METHOD: GET
SESSION ID:
GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  Host: 10.148.118.211:7001
  Connection: keep-alive
  Referer: http://10.148.118.211:7001/pagelets/bidwiki/dashboard.action
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.16
(KHTML, like Gecko) Chrome/10.0.648.205 Safari/534.16
  Accept: */*
  Accept-Encoding: gzip,deflate,sdch
  Accept-Language: en-US,en;q=0.8,ru;q=0.6,it;q=0.4
  Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
  If-None-Match: W/"736-1124953206000"
  If-Modified-Since: Thu, 25 Aug 2005 07:00:06 GMT
COOKIES:
  JSESSIONID: GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614
```

Example 63–6 HTTP Response Sent By Pagelet Producer

```
URL: http://10.148.118.211:7001/pagelets/bidwiki/styles/main-action.css
Response Code: 200
Reason:
HEADERS:
  Date: Thu, 19 May 2011 01:39:14 GMT
  Content-Type: text/css;charset=UTF-8
  Server: Apache-Coyote/1.1
BODY:
.sidebar {
    /*background-image: url(http://10.148.118.211:7001/pagelets/bidwiki/download/
resources/leftnav_bg.jpg);*/
    /*background-repeat: repeat-y;*/
    background-color: #F0F0F0;
    /*border-bottom:1px solid #F0F0F0;*/
} ...
```

Example 63–7 HTTP Request Sent By Pagelet Producer

```
URL: http://bidwiki.us.oracle.com/includes/js/ajax.js
METHOD: GET
HEADERS:
  CSP-Ensemble-REST-API: http://10.148.118.211:7001/pagelets
  X-Client-IP: 10.148.118.211
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.16
(KHTML, like Gecko) Chrome/10.0.648.205 Safari/534.16
  CSP-Session-Username: weblogic
  Accept-Language: en-US,en;q=0.8,ru;q=0.6,it;q=0.4
  CSP-Gateway-Type: Proxy
  PT-Proxy-Passes: 1
  If-Modified-Since: Thu, 25 Aug 2005 07:00:06 GMT
  CSP-Protocol-Version: 1.4
  Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
  Accept-Encoding: gzip,deflate,sdch
  Referer: http://10.148.118.211:7001/pagelets/bidwiki/dashboard.action
  If-None-Match: W/"736-1124953206000"
  Accept: */*
  CSP-Aggregation-Mode: Single
  PT-Proxy-instance0: {DECBB085-D891-72CF-2B75-005E7FE20000}
  CSP-Gateway-Specific-Config: PT-User-Name=weblogic,PT-Guest-User=0,...
```

Example 63–8 HTTP Response Received By Pagelet Producer

```
Original URI: http://bidwiki.us.oracle.com/styles/main-action.css
Effective URI: http://bidwiki.us.oracle.com/styles/main-action.css
Status Code: 200
Reason: OK
Version: HTTP/1.1
HEADERS:
  Content-Type: text/css;charset=UTF-8
  Content-Length: 29178
  Server: Apache-Coyote/1.1
  Date: Thu, 19 May 2011 01:39:14 GMT
TRAILERS:
BODY:
body, p, td, table, tr, .bodytext, .stepfield {
  font-family: Verdana, arial, sans-serif;
  font-size: 11px;
  line-height: 16px;
  color: #000000;
```

```
  font-weight: normal;
}
```

63.3.4.2 Transformation Tracing

To view the content proxied by the Pagelet Producer before and after transformation, set the **Transform** component on the Logging Settings page to **Finest**. The purpose of these traces is to log response content at different stages of transformation, allowing you to compare them and view the result of different transformers.The example traces below were captured from a test environment.

Example 63–9 Untransformed Markup

```
Original request: URL:
http://10.148.118.211:7001/pagelets/bidwiki/styles/main-action.css
METHOD: GET
SESSION ID:
GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  Host: 10.148.118.211:7001
  Connection: keep-alive
  Referer: http://10.148.118.211:7001/pagelets/bidwiki/dashboard.action
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.16
(KHTML, like Gecko) Chrome/10.0.648.205 Safari/534.16
  Accept: text/css,*/*;q=0.1
  Accept-Encoding: gzip,deflate,sdch
  Accept-Language: en-US,en;q=0.8,ru;q=0.6,it;q=0.4
  Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
COOKIES:
  JSESSIONID: GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614

Untransformed content:
body, p, td, table, tr, .bodytext, .stepfield {
  font-family: Verdana, arial, sans-serif;
  font-size: 11px;
  line-height: 16px;
  color: #000000;
  font-weight: normal;
}
```

Example 63–10 Transformed Markup (Transformed by Transformer Class)

```
Transformed by: class com.plumtree.server.impl.portlet.transformers.CSSTurboParser
  Original request: URL:
http://10.148.118.211:7001/pagelets/bidwiki/styles/main-action.css
METHOD: GET
SESSION ID:
GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614!1305769149498
HEADERS:
  Host: 10.148.118.211:7001
  Connection: keep-alive
  Referer: http://10.148.118.211:7001/pagelets/bidwiki/dashboard.action
  User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/534.16
(KHTML, like Gecko) Chrome/10.0.648.205 Safari/534.16
  Accept: text/css,*/*;q=0.1
  Accept-Encoding: gzip,deflate,sdch
  Accept-Language: en-US,en;q=0.8,ru;q=0.6,it;q=0.4
  Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
COOKIES:
  JSESSIONID: GdYGNJzMhxy1CJBMVTX8xTNq32GmLXYNY9VqFBcdprFnhcyQtzdp!1377086614
```

```
Transformed content:
body, p, td, table, tr, .bodytext, .stepfield {
  font-family: Verdana, arial, sans-serif;
  font-size: 11px;
  line-height: 16px;
  color: #000000;
  font-weight: normal;
}
```

63.4 Working with WSRP and Oracle JPDK Portlets

The Pagelet Producer can be used to present WSRP and Oracle JPDK portlets for use in any web application. For details on configuring the Pagelet Producer to connect to a WSRP or Oracle JPDK portlet producer, see *Managing the Oracle WebCenter Portal's Pagelet Producer* in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. The Pagelet Producer can also modify portlet markup at runtime as explained in the next section, Section 63.4.1, "Using Pagelet Chrome."

63.4.1 Using Pagelet Chrome

Pagelet chrome allows you to modify markup at runtime in pagelets created from portlets imported from a WSRP or JPDK portlet producer.

A pagelet chrome template is an HTML file that specifies:

- How portlet markup should be rendered stylistically
- How to display the portlet title
- How to render mode switching options

The default chrome template displays the portlet name and a dropdown menu that allows the user to switch into different modes. The dropdown menu is dynamically populated with all the standard modes supported by the underlying portlet. Templates uses the following reserved tokens to identify key portlet elements, which the Pagelet Producer substitutes at runtime.

Table 63–3 Pagelet Chrome Template Tokens

Token	Description
`$$PORTLET TITLE$$`	Portlet title
`$$REPEAT MENU ITEM$$`	Used to indicate the beginning of a repeating section for navigational items
`$$END REPEAT MENU ITEM$$`	Used to indicate the end of a repeating section for navigational items
`$$MENU ITEM URL$$`	Navigation URL (to switch portlet modes or window states)
`$$MENU ITEM$$`	Display name of navigational item (for example, Customize)
`$$TOKEN$$`	Unique identifier for pagelet instance on the page
`$$PORTLET CONTENT$$`	Portlet content
`pt://images`	Tag used to indicate the imageserver URL

The example below is a very simple pagelet chrome template:

```
<script type="text/javascript">
function goto(url)
```

```
{
  document.location = url;
  return false;
}
</script>
<div style="border: 1px solid">
<span><b><!-- $$PORTLET TITLE$$ --></b></span>
<span style="align: right">
  Switch Mode:
  <select size="1" name="mode">
    <!-- $$REPEAT MENU ITEM$$ -->
    <option onclick="goto('$$MENU ITEM URL$$')"><!-- $$MENU ITEM$$ --></option>
    <!-- $$END REPEAT MENU ITEM$$ -->
  </select>

</span>
</div>
<!-- $$PORTLET CONTENT$$ -->
```

Note: The pagelet chrome template file must be hosted on the classpath of the Pagelet Producer web application.

If you configured Pagelet Producer to use an external image server, copy the files from ensemblestatic.war/imageserver/yahoo to your image server to properly render the default chrome template.

To implement the chrome template, add it as a parameter to the pagelet inject URL (REST or JavaScript). For details on pagelet inject URLs, see Section 63.2.2, "Adding a Pagelet to a Web Page." For example:

- REST: `/inject/v2/pagelet/pagelet_lib/pagelet_name?chrome=mychrome.html`
- JavaScript: `injectpagelet(library, name, iframe_options, payload, params, context_id, element_id, is_in_community, chrome)`

The value of the chrome parameter can be the name of the file containing the chrome template or the special reserved value "`none`", which suppresses all chrome and sends back portlet markup only. If the chrome parameter is omitted, the default chrome is returned with the portlet markup. The default chrome template uses YUI menu control to display a gradient title bar and a DHTML dropdown menu for switching modes. (When ADF content is detected, a different chrome template is used by default. This template can be overridden with a custom template or with the standard default template by setting chrome=chrometemplate.html.)

63.5 Working with OpenSocial Gadgets

Any OpenSocial gadget reachable by the Pagelet Producer server can be registered as a pagelet and used in any web application, including a portal. For details on registering OpenSocial pagelets with the Pagelet Producer, see *Managing the Oracle WebCenter Portal's Pagelet Producer* in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The Pagelet Producer supports most of the standard OpenSocial APIs excluding OAuth. The complete OpenSocial API reference documentation can be found here: `http://shindig.apache.org/shindig-1.1.x/shindig-features/jsdoc/index.html`.

The Pagelet Producer also allows gadgets to store preferences, retrieve WebCenter Portal profile and connection information, and access a user's activity stream using OpenSocial APIs. For more details on using these features, see the sections that follow.

63.5.1 Configuring Authentication

In order for gadgets to request user-level data (preferences or people connections), the end user's identity must be established. If any OpenSocial gadgets need to access user-level data from the server, you must configure a security policy for the parent OpenSocial resource in the Pagelet Producer Console. The first time a user accesses an OpenSocial gadget, a login page will be presented. After the initial login, subsequent requests for OpenSocial gadgets will use the established user identity.

For details on configuring a security policy for a Pagelet Producer resource, see the Managing Pagelets chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*

63.5.2 Storing User Preferences

OpenSocial gadgets may use user preferences to store data at the container. User preferences are scoped to a particular user and may optionally be scoped to an appId (the gadget appId is the pagelet context ID). If you choose to use the OpenSocial gadget.Prefs API, the user preferences will be scoped to the user and pagelet instance. Alternatively, you may use the opensocial.DataRequest API to manage preferences at the user level that can be shared with other pagelets.

When registered as a pagelet, a gadget's user preferences are treated as pagelet preferences. In WebCenter Portal, for example, non-hidden user preferences can be edited by the end user using the Personalize button. Additionally, to simulate preferences shared between users, you may pass in user preferences via pagelet parameters. Note that a pagelet preference, if set, will always override the corresponding pagelet parameter (in other words, personalization takes precedence over customization).

Outside of WebCenter Portal, gadget-backed pagelets are provided with a simple chrome that displays the gadget title and buttons for accessing the preference editor and minimizing/maximizing the gadget. The chrome may be suppressed by passing in the value of 'none' to the chrome parameter in the Pagelet Inject API. The preference editor UI supports all four types of user preferences:

- string: rendered as a text field
- bool: rendered as a checkbox
- enum: rendered as a dropdown list
- list: rendered as a text field (values must be separated with a pipe "|" character)

63.5.3 Accessing WebCenter Portal Profile Information

OpenSocial gadgets can query the current user's profile data and people connections via the standard OpenSocial APIs. To use this feature, you must manually target the WebCenterDS data source to the WC_Portlet server as described in *Managing the Oracle WebCenter Portal's Pagelet Producer* in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Note: The OpenSocial API cannot be used to update profile or connection information.

The supported user profile fields are listed in the table below.

Table 63–4 User Profile Fields

OpenSocial Field	Type	Description
aboutme	string	A general statement about the person.
addresses	Plural-Field <Address>	A physical mailing address for the person.
appData	Plural-Field <AppData>	A collection of AppData keys and values, used for preferences.
birthday	Date	The birthday of the person. The value MUST be a valid Date. The year may be set to 0000 when the age of the person is private or the year is not available.
emails	Plural-Field <string>	E-mail address for the person.
location	string	Physical address for the person.
name	Name	The broken-out components and formatted version of the person's real name.
organizations	Plural-Field <Organization>	Current or past organizational affiliation of the person.
phoneNumbers	Plural-Field <string>	Phone number for the person. In addition to the standard canonical values for type, this field defines the additional values mobile, fax, and pager.
photos	Plural-Field <string>	URL to a photo of the person. The value MUST point to an actual image file (e.g. a GIF, JPEG, or PNG image file) rather than to a web page containing an image. Note that this field SHOULD NOT be used to send down arbitrary photos taken by this user, but specifically profile photos of the contact suitable for display when describing the contact.
preferredUsername	string	The preferred username of this person on sites that ask for a username (e.g. jsmarr or daveman692).
status	string	The person's status, headline or shoutout.
thumbnailUrl	string	The person's photo thumbnail URL, specified as a string. This URL MUST be fully qualified.

63.5.4 Accessing a User's Activity Stream

OpenSocial gadgets can operate on a user's activity stream using OpenSocial APIs. The following operations are supported:

- get activities
- create activity

The following operations are not supported:

- update activity
- delete activity

The supported activity stream fields are listed in the table below.

Table 63–5 Activity Stream Fields

OpenSocial Field	Type	Description
appId	Object-Id	The application with which the activity is associated.
body	string	An optional expanded version of the activity. Bodies may only have the following HTML tags: <b> <i>, <a>, , but this formatting may be ignored.
externalId	Object-Id	An optional ID generated by the posting application.
id	Object-Id	An ID that is permanently associated with the activity.
postedTime	string	The time at which the activity took place in milliseconds since the epoch.

Table 63–5 (Cont.) Activity Stream Fields

OpenSocial Field	Type	Description
priority	number	A number between 0 and 1 representing the relative priority of the activity in relation to other activities from the same source.
title	string	The primary text of the activity. Titles may only have the following HTML tags: <b> <i>, <a>, , but this formatting may be ignored.
userId	Object-Id	ID of the user for whom the activity is defined.

63.5.5 Using Gadget Eventing

The Pagelet Producer supports the OpenSocial pubsub inter-gadget eventing model. A gadget may publish events over any number of arbitrary channels (defined by simple string names) in JavaScript. On the receiving end, a gadget may subscribe to receive events over any number of channels, again in JavaScript, and take appropriate actions based on the events.

```
function connSelected(element) {
  var connId = element.id;
  var connName = element.lastChild.textContent;
  gadgets.pubsub.publish(channel, {id: connId, name: connName});
```

For a complete JavaScript reference to the supported eventing API, see http://shindig.apache.org/shindig-1.1.x/shindig-features/jsdoc/symbols/gadgets.pubsub.html.

63.5.6 Example: Consuming an External OpenSocial Gadget

The example that follows is simplified for illustration. For details on Pagelet Producer Console settings, see the *Managing Oracle WebCenter Portal's Pagelet Producer* chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

1. Using the Pagelet Producer Console, create a new Resource and choose **OpenSocial** as the type. Define URLs and policies as necessary. Save the new Resource.
2. Using the Pagelet Producer Console, create a new pagelet and enter the location of the gadget XML schema in the **Gadget URL** field.
   - To import the gadget name and any user preferences defined for the gadget, click the **Import Gadget Metadata** button. If any of the preferences are editable, the Pagelet Producer will create a preference editor using the imported preferences.
   - To configure or customize the generated preference editor, go to the **Pagelet Parameters** page.
3. Go to the **Documentation** page to view sample code to access the pagelet and preference editor using either the JavaScript or REST API.
4. Save the new pagelet.

63.5.7 Example: Consuming a Local OpenSocial Gadget

The example that follows is simplified for illustration. For details on Pagelet Producer Console settings, see the *Managing Oracle WebCenter Portal's Pagelet Producer* chapter in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

1. Using the Pagelet Producer Console, create a new Resource and choose **OpenSocial** as the type. Define URLs and policies as necessary. Save the new Resource.

2. Create or upload the gadget file.

   Note: If you choose to host OpenSocial gadget XML files, the files must be placed under an anonymous resource (with no security policy) or gadget functionality will not work correctly.

   a. Select the **Files** section under the resource and click the **Create** icon in the toolbar.

   b. On the **General** page, enter the relative path to the file in the **Name** field. Do not use a leading forward-slash ("/"). For example, "gadgets/activities.xml". Any path and name can be used; the path is a 'virtual' URL that the Pagelet Producer will use to serve the file. The visual directory structure of the Files collection in the navigator is updated to match the path to the file.

   c. Go to the **Content** page to upload or create the gadget file.

      To upload a gadget file, enter the path to the file or click the Browse button to navigate to the file, then click the **Upload** button to upload the file to the Pagelet Producer server.

      To create a gadget file, use the editor on the **Content** page to enter and edit content.

   d. Click the **Save** icon in the navigator toolbar.

3. Create or upload any other files required by the gadget (JavaScript, images, etc.) in the **Files** section of the resource. Define paths for the files that match the paths in the gadget code. For example, if the gadget uses JavaScript files in a subfolder called "js", include that directory in the Name field when you upload the files, e.g., "gadgets/js/activities.js".)

4. Using the Pagelet Producer Console, create a new pagelet and enter the relative path to the gadget XML schema (created or uploaded in step 2) in the **Gadget URL** field. For local files, this is the same path defined in the Name field for the file object; for example, "gadgets/activities.xml".

   - To import the gadget name and any user preferences defined for the gadget, click the **Import Gadget Metadata** button. If any of the preferences are editable, the Pagelet Producer will create a preference editor using the imported preferences.
   - To configure or customize the generated preference editor, go to the **Pagelet Parameters** page.

5. Go to the **Documentation** page to view sample code to access the pagelet and preference editor using either the JavaScript or REST API.

6. Save the new pagelet.

64

Consuming Portlets

This chapter describes how to add portlets to the pages of your WebCenter Portal: Framework application and the options that accompany this process.

This chapter includes the following sections:

- Section 64.1, "Introduction to Consuming Portlets"
- Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application"
- Section 64.3, "Adding Portlets to a Page"
- Section 64.4, "Setting Attribute Values for the Portlet Tag"
- Section 64.5, "Copying Portlets"
- Section 64.6, "Deleting Portlets from Application Pages"
- Section 64.7, "Consuming WebCenter Services Portlets"
- Section 64.8, "Adding the Portlet Producer Task Flow"
- Section 64.9, "Troubleshooting Portlets"

This chapter does not cover Oracle JDeveloper or Oracle ADF page creation basics. It covers only those aspects of page creation that are specific to Framework application pages. Therefore, you should familiarize yourself with the information covered in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework* before reading this chapter.

For information about creating portlets, which can then be consumed by Framework application pages, see the following chapters:

- Chapter 58, "Overview of Portlets"
- Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge"
- Chapter 60, "Creating Portlets with the Portlet Wizard"
- Chapter 65, "Creating Portlets with OmniPortlet"
- Chapter 66, "Creating Content-Based Portlets with Web Clipping"

64.1 Introduction to Consuming Portlets

Oracle WebCenter Portal: Framework enables you to consume a portlet by registering its producer either with an application or with the Resource Palette from where you can add it to any application. After you register the producer, its portlets appear under the registered producer's name under the Connections node in the Application Resources panel or in the Resource Palette.

Your application can consume portlets that you build and portlets that you receive from a third party, such as a packaged-application vendor.

There are many options associated with portlet consumption. For example, you can choose to place portlets straight onto a page or nest them in a Composer component, you can adjust many attributes of the portlet tag, and you can wire portlets to each other.

64.2 Registering Portlet Producers with a WebCenter Portal: Framework Application

Before you can add a portlet to a Framework application page, you must register the portlet's producer with the application. You can register portlet producers in two ways:

- Register the portlet producer with a specific application. Use this option if you are not likely to want to register the producer with other applications.
- Register the portlet producer using the Resource Palette. Use this option to use the producer's portlets in multiple applications.

A portlet that is available in the Resource Palette can be added to any of your Framework applications by dropping it on the page as you would any other component. When you add a portlet from the Resource Palette, its producer gets registered with the application if that producer is not already registered with the application. You can drag and drop a whole producer connection from the Resource Palette into the Application Resources panel of the Application Navigator. This registers the producer with the application. Alternatively, you can right-click a producer in the Resource Palette and choose **Add to Application** from the context menu to register the producer with the currently open application.

JDeveloper provides wizards for registering both WSRP producers and Oracle PDK-Java producers.

Note: If your application is source controlled, you must manually create elements in the source control system for any new files created during producer registration. Any files that are already source controlled are checked out automatically by the producer registration process.

This section includes the following subsections:

- Section 64.2.1, "How to Register a WSRP Portlet Producer"
- Section 64.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles"
- Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer"
- Section 64.2.4, "How to Edit Portlet Producer Registration Settings"
- Section 64.2.5, "How to Test a Portlet Producer Connection"
- Section 64.2.6, "How to Refresh a Portlet Producer"
- Section 64.2.7, "How to Delete a Portlet Producer"

For more information about producers, see Section 58.2.3, "Deployment Type." For information about obtaining prebuilt portlets through Oracle, see Section 3.4, "Working with the Integrated WebLogic Server." For information about using

JDeveloper's portlet creation wizards, see Chapter 60, "Creating Portlets with the Portlet Wizard." For more information about portlets, see Chapter 58, "Overview of Portlets."

64.2.1 How to Register a WSRP Portlet Producer

When you register a WSRP portlet producer, you provide basic information that describes the producer's operational parameters. This information is used by the portlet-consuming application to communicate with the producer and with the portlets through the producer.

WebCenter Portal: Framework supports both WSRP 1.0 and WSRP 2.0 producers. The WSRP 2.0 standard, among others, provides support for inter-portlet communication and export and import of portlet customizations. You can leverage the benefits of WSRP 2.0 while building standards-based JSR 286 portlets.

The Register WSRP Portlet Producer wizard is the entry point for registering both WSRP 1.0 and 2.0 producers. When registration is successful, the newly registered producer displays in JDeveloper either in the Application Resources panel of the Application Navigator, or in the Resource Palette, depending on where you created the connection. You can then select portlets from the producer for placement on your application (`.jspx`) pages.

Note: If you are registering a producer provided by Oracle WebLogic Portal, and the portlet uses ADF Rich Components, you should register the WSRP 2.0 WSDL URL to ensure that the portlet functions correctly.

You also use the Register WSRP Portlet Producer wizard to register JSF portlets, which are portletized JSF applications or portletized ADF task flows. Once you create a portlet from a JSF application, you can deploy the portlet to a WLS instance and register the JSF portlet producer as you would register any WSRP portlet producer. The Oracle JSF Portlet Bridge exposes JSF applications and task flows as JSR 286 portlets. For more information, see Chapter 59, "Creating Portlets with the Oracle JSF Portlet Bridge."

To register a WSRP portlet producer:

Note: In the Register WSRP Portlet Producer wizard, if you click **Cancel** after you have clicked **Finish,** the registration is not canceled.

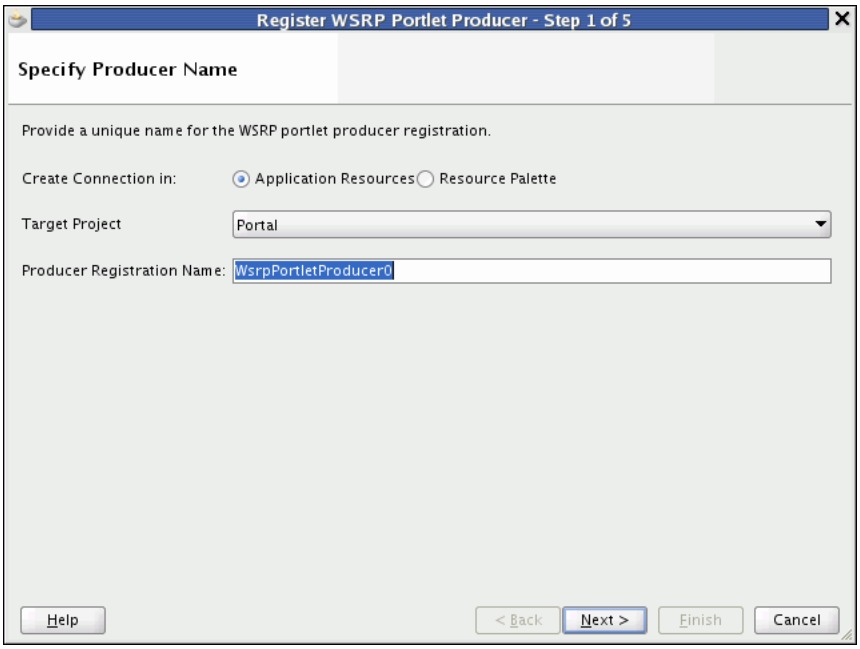
---

1. In the Application Resources panel of the Application Navigator, right-click **Connections,** choose **New Connection** and then choose **WSRP Producer.**

   For other methods of invoking the wizard, such as from the Resource Palette, see Section 1.6.2, "How Do I Access the Connection Wizards?"

2. In the Specify Producer Name page (Figure 64–1) of the Register WSRP Portlet Producer wizard, the **Create Connection in** option is set depending on how you accessed the wizard. The default selection is **Application Resources** if you invoked the wizard from an application, and **Resource Palette** if you invoked the wizard from the Resource Palette. You can change this option at this point.

Figure 64–1 The Specify Producer Name Page (WSRP Producer)



3. From the **Target Project** dropdown list, select the project to be configured for the WSRP producer connection.

   This should be the same project as the one in which you intend to consume the portlets.

   You can change this option only if you invoked the wizard from the Application Navigator.

4. In the **Producer Registration Name** field, enter a name for the producer registration that is unique among all connections and then click **Next.**

5. In the Specify Connection Details page (Figure 64–2), in the **WSDL URL** field, enter the producer's URL.

   The syntax varies according to your WSRP implementation, for example, the sample WSRP producer uses the following syntax:

   - `http://`*`host`*`:`*`port`*`/`*`context-root`*`/portlets/wsrp1?WSDL`
   - `http://`*`host`*`:`*`port`*`/`*`context-root`*`/portlets/wsrp2?WSDL`
   - `http://`*`host`*`:`*`port`*`/`*`context-root`*`/portlets?WSDL` (WSRP 1.0 for backward compatibility)

   Where:

   - *`host`* is the server to which your producer has been deployed.
   - *`port`* is the port to which the server is listening for HTTP requests.
   - *`context-root`* is the Web application's context root.
   - `portlets[/wsrp(1|2)]?WSDL` is static text. The text entered here depends on how the producer is deployed.

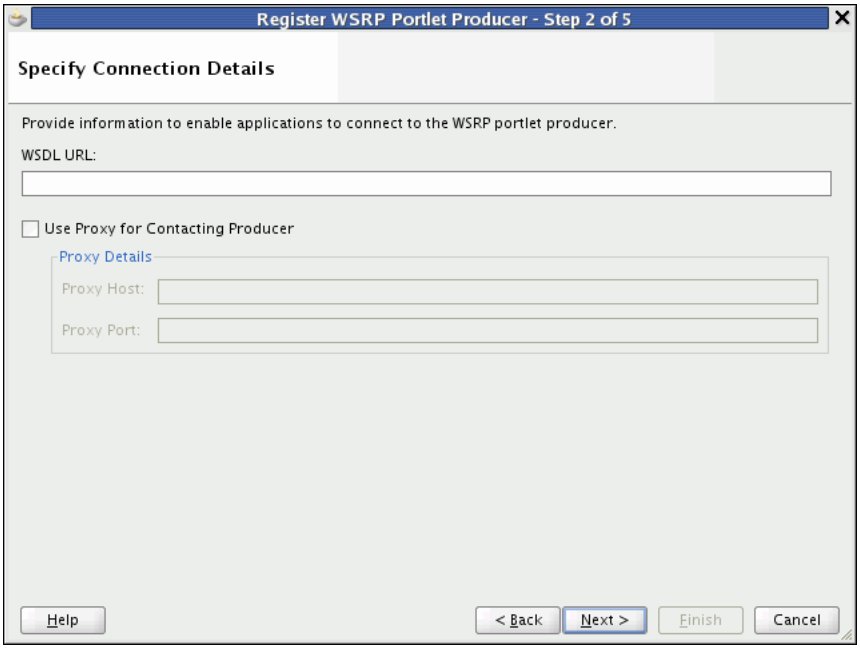
   For example:

   ```
   http://myhost.example.com:7101/portletapp/portlets/wsrp2?WSDL
   ```

You can access the producer test page through the URL:

`http://host:port/context-root/info`

Figure 64–2 The Specify Connection Details Page (WSRP Producer)



6. If the application and the producer are separated by a firewall, an HTTP proxy is needed for communication between the application and the producer. If this is the case for your application, select **Use Proxy for Contacting Producer** and specify the URL and port number of the proxy.

   Note: The proxy fields in this step default to the proxy preferences set in JDeveloper Preferences (from the main menu, choose **Tools > Preferences,** and then select **Web Browser and Proxy**).

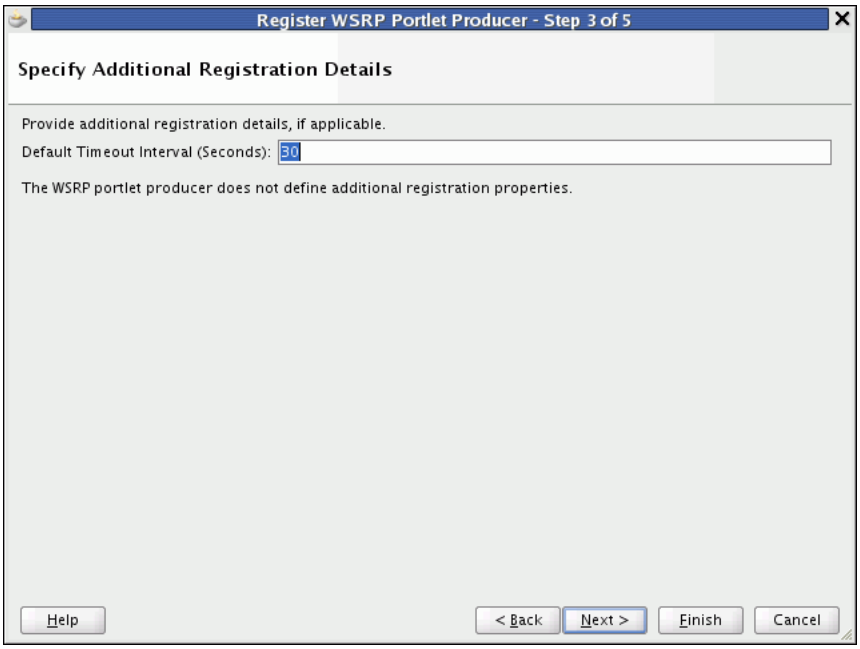
7. Click **Next.**

   The connection to the producer is tested. If there are any problems, an error message displays. You must resolve any problems before you can continue.

8. In the Specify Additional Registration Details page (Figure 64–3), in the **Default Timeout Interval (Seconds)** field, enter the number of seconds to wait for the producer to respond during design time operations.

   Some producers define additional registration properties. In such cases, the properties are displayed in a table on this page of the wizard. You can enter values for these additional properties in the table. These properties are producer-specific and are used only at registration time. That is, they collect information that consumer applications send to producers at registration time; the producers store this information against the consumers and use it subsequently.

Figure 64–3 The Specify Additional Registration Details Page (WSRP Producer)



9. If you are registering the producer in the Resource Palette, click **Finish** to complete the registration.

   If you are registering the producer in the Application Resources panel, and plan to request authentication whenever the producer (and consequently, its portlet) is accessed, click **Next** and follow the remaining steps. If you do not want to configure security, click **Finish.**

   If the producer declares user categories, when you click **Finish,** the Register WSRP Portlet Producer dialog displays. Click **Yes** and see Section 64.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles." Click **No** to decline this opportunity and complete the registration process.

   Note: If you decline to map the user categories to security roles at this point, you can do so later by editing the producer registration.

10. In the Configure Security Attributes page (Figure 64–4), from the **Token Profile** dropdown list, select the type of token profile to use for authentication with the WSRP producer:

    - **None**—No token; no WS-Security header is attached to the SOAP message.

      If you select this option, you do not want to complete the rest of the wizard. Click **Finish.**

    - **WSS 1.0 SAML Token with Message Integrity**—Provides message-level integrity protection and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation. This policy uses WS-Security's Basic 128 suite of asymmetric key technologies and SHA-1 hashing algorithm for message integrity.

- **WSS 1.0 SAML Token with Message Protection**—Provides message-level protection (integrity and confidentiality) and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. The Web service consumer includes a SAML token in the SOAP header and the confirmation type is sender-vouches. This policy uses WS-Security's Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.

  When you select this policy, you must also specify the Recipient Alias.

- **WSS 1.0 User Name Token without Password**—Provides username (with password) token profile based identity propagation with certificate based message protection for outbound SOAP requests in accordance with the WS-Security v1.0 standard. Both plain text and digest mechanisms are supported. This policy uses WS-Security's Basic128 suite of asymmetric key technologies. Specifically, RSA key mechanism for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.

  Use this token profile if the WSRP producer has a different identity store. You must define an external application pertaining to the producer and associate the external application with this producer. The external application defined here is used to retrieve and propagate the user credentials to the producer. The producer verifies this against the identity store configured for the external application.

  When you select this policy, you must also specify the Recipient Alias.

- **WSS 1.0 User Name Token with Password**—provides username (with password) token profile based identity propagation with certificate based message protection for outbound SOAP requests in accordance with the WS-Security 1.0 standard. Credentials (username only) are included in outbound SOAP request messages through a WS-Security UsernameToken header. No password is included. Message protection is provided using WS-Security 1.0's Basic128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.

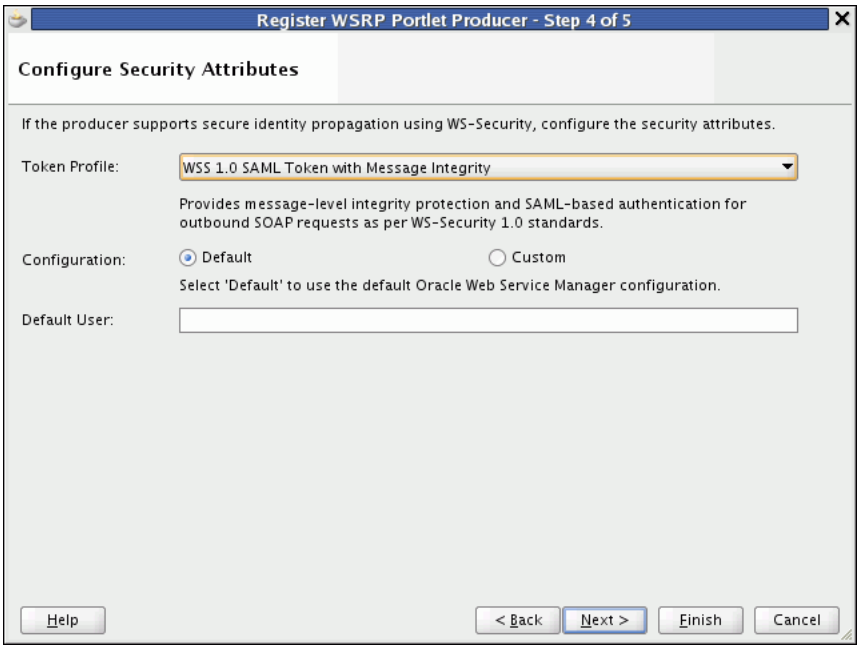
  When you select this policy, you must also specify the Recipient Alias.

- **WSS 1.0 SAML Token**—Provides SAML-based authentication for outbound SOAP request messages in accordance with the WS-Security 1.0 standard. The policy propagates user identity and is typically used in intra departmental deployments where message protection and integrity checks are not required.

  This policy does not require any keystore configuration.

- **WSS 1.1 SAML Token with Message Protection**—Provides message-level protection (integrity and confidentiality) and SAML token population for outbound SOAP requests in accordance with the WS-Security 1.1 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation. This policy uses the symmetric key technology for signing and encryption, and WS-Security's Basic128 suite of asymmetric key technologies for endorsing signatures.

Figure 64–4 The Configure Security Attributes Page (WSRP Producer)



11. Select the configuration type:

    - **Default**—If you choose default, then all the default keystore attributes, that is location, password, keystore type, signature key and alias, encryption key and alias are picked up from the JPS (Java Platform Security) configuration. The value for recipient alias is used from the policy being used. The WebLogic Server where the application is deployed must be configured for WS-Security. For more information, see "Securing a WSRP Producer with WS-Security" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
    - **Custom**—If you select this option, then you must enter the appropriate keystore attributes in the next page of the wizard.

12. In the **Default User** field, enter a user name to assert to the remote producer when the user has not authenticated to the Framework application.

    When unauthenticated, the identity *anonymous* is associated with the application user. OWSM does not currently support the propagation of an anonymous identity, so you must specify an alternative identity here. Keep in mind though, that in this case, the Framework application has not authenticated the user so the default user you specify should be a low privileged user in the remote producer that is an appropriate identity to use for showing public content. For example, you may want to create a guest account in the identity store for this purpose. If the user has authenticated to the application, then the user's identity is asserted rather than the default user.

    > **Note:** If you specify a **Default User**, the remote producer must be set up to accept this information.

    The **Default User** field does not appear if you selected **User Name Token with Password.**

13. In the **Issuer Name** field, enter the name of the issuer of the SAML Token, for example `www.oracle.com`.

This field appears only if you selected an **SAML Token** option from the **Token Profile** dropdown list, and **Custom** from the **Configuration** options. The issuer name is the attesting entity that vouches for the verification of the subject.

14. Select **Associate Producer with External Application**, then select the application, if this producer must provide authentication to an external application.

    For more information, see Section 69.13.3, "Managing External Applications."

    This option is available only if you selected **User Name Token with Password.**

15. Click **Next.**

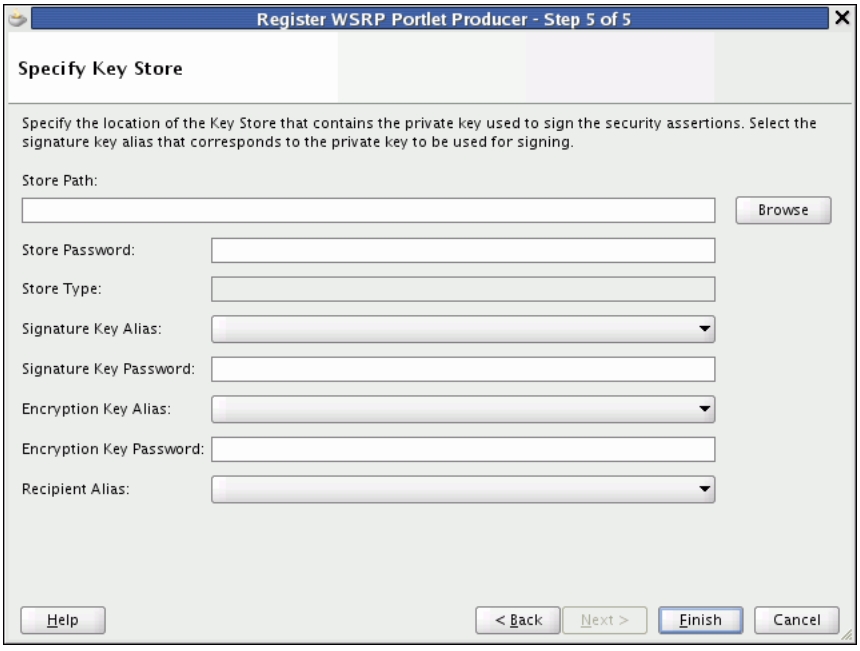
    If you selected **Default** as the configuration option, then the fields on the Specify Key Store page are disabled. Click **Finish** to complete the registration.

    If the producer declares user categories, when you click **Finish,** the Register WSRP Portlet Producer dialog displays. Click **Yes** and see Section 64.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles." Click **No** to decline this opportunity and complete the registration process.

16. In the Specify Key Store page, in the **Store Path** field, provide the full path to the keystore that contains the certificate and the private key that is used for signing some parts (security token and SOAP message body) of the SOAP message.

    If you are not sure of the full path, click **Browse** to navigate to and select the file. The selected file should be a keystore created with the Java keytool.

Figure 64–5 The Specify Key Store Page (WSRP Producer)



17. In the **Store Password** field, provide the password to the keystore that was set when the keystore was created.

    The keystore password must be correct for the **Store Type** field and the **Signature Key Alias** dropdown list to populate.

    If an incorrect keystore path or password is entered, then an error message appears stating that the password is invalid and must be corrected. All fields on

this screen except for **Store Path** and **Store Password** are disabled until you specify the correct values.

18. After you provide the correct keystore path and password, press the Tab key to move to another active field (for example, the **Store Path** field). This ensures that the **Store Type** field and the **Signature Key Alias** dropdown list are properly populated.
19. The **Store Type** value is read from the keystore and is never editable. The store type is always **JKS** (Java Key Store).
20. From the **Signature Key Alias** dropdown list, select the signature key alias.

    This list populates automatically when the correct password is entered in the **Store Password** field. The **Signature Key Alias** is the identifier for the certificate associated with the private key that is used for signing. The key aliases found in the specified keystore are available in the list. Select the one to be used for signing.
21. In the **Signature Key Password** field, specify the password for accessing the key identified by the alias specified in **Signature Key Alias.**
22. Optionally, from the **Encryption Key Alias** dropdown list, select the encryption key alias.

    This list populates automatically when the correct password is entered in the **Store Password** field. The key aliases found in the specified keystore are available in the list. Select the one to be used for encryption.
23. Optionally, in the **Encryption Key Password** field, specify the password for accessing the key identified by the alias specified in **Encryption Key Alias**.
24. From the **Recipient Alias** field, select the keystore alias that is associated with the producer's certificate and then click **Finish.**

    This certificate is used to encrypt the message to the producer.

    This field is not displayed if you selected **SAML Token with Message Integrity** as the **Token Profile** in the Configure Security Attributes page of the wizard.

    If the producer declares user categories, when you click **Finish,** a dialog displays asking whether you want to map the user categories to Java EE roles. Click **Yes** and see Section 64.2.2, "How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles." Click **No** to decline this opportunity and complete the registration process.

64.2.2 How to Map a Producer's Declared User Categories to an Application's Defined Java EE Security Roles

The user categories the producer declares come from the portlets it contains. For example, if the producer contains one or more JSR 286 portlets created with the Standards-based Java Portlet (JSR 286) Wizard, then any security roles added during portlet creation are included in the user categories the producer declares. Java EE Security Roles can be specified through the Framework application's `web.xml` file properties.

For more information about security roles in JSR 286 portlets, see Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security."

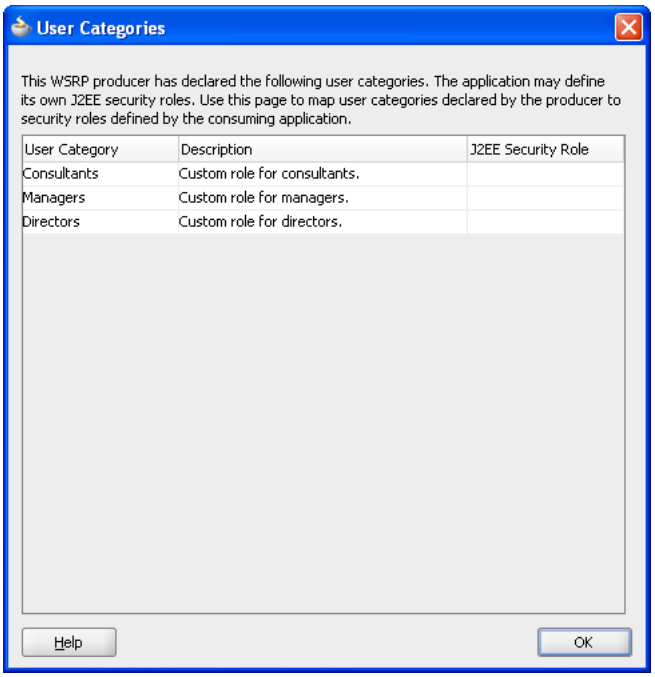
This procedure continues forward from Section 64.2.1, "How to Register a WSRP Portlet Producer."

To map producer-declared user categories with application-defined Java EE security roles:

1. After clicking **Finish** in the Register WSRP Portlet Producer wizard, click **Yes** in the resulting dialog.
2. In the User Categories dialog (Figure 64–6), for each **User Category**, click the corresponding field in the **J2EE Security Role** column.

   The User Categories dialog is also accessible when you edit producer registration settings. For more information see Section 64.2.4, "How to Edit Portlet Producer Registration Settings."

Figure 64–6 The User Categories Dialog



3. From the resulting list, select the security role to map to the producer user category.
4. Click **OK** when all user categories are mapped.

64.2.3 How to Register an Oracle PDK-Java Portlet Producer

When you register a PDK-Java portlet producer, you provide basic information that describes the producer's operational parameters. This information is used by the portlet-consuming application to communicate with the producer and with the portlets through the producer.

When registration is successful, the newly registered producer is displayed in JDeveloper either in the Application Resources panel of the Application Navigator, or in the Resource Palette, depending on where you created the connection. You can then select portlets from the producer for placement on your application (`.jspx`) page.

> **Note:** In the Register Oracle PDK-Java Portlet Producer wizard, if you click **Cancel** after you have clicked **Finish,** the registration is not canceled.

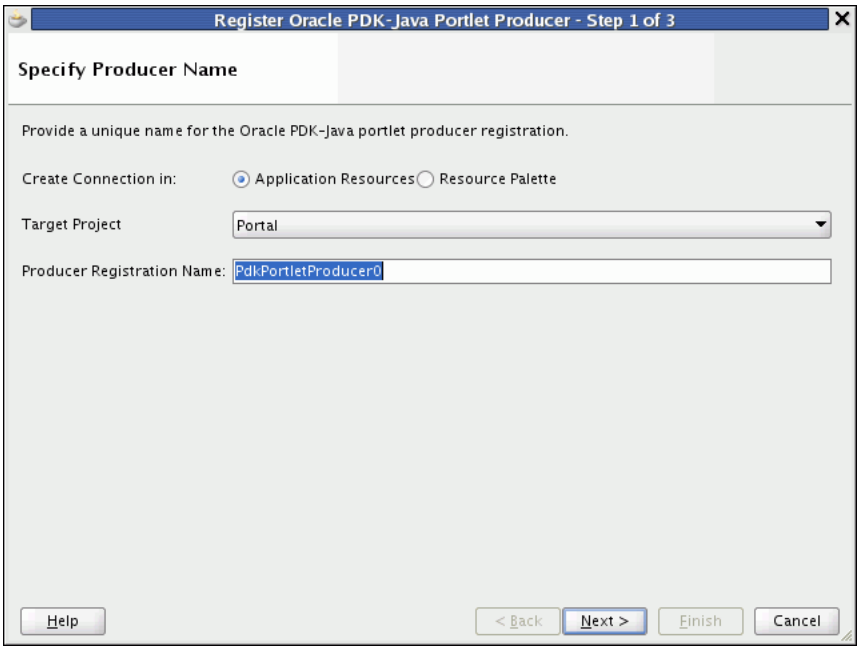
To register a PDK-Java portlet producer:

1. In the Application Resources panel of the Application Navigator, right-click **Connections,** choose **New Connection** and then choose **Oracle PDK-Java Producer.**

   For other methods of invoking the wizard, such as from the Resource Palette, see Section 1.6.2, "How Do I Access the Connection Wizards?."

2. In the Specify Producer Name page (Figure 64–7) of the Register Oracle PDK-Java Portlet Producer wizard, the **Create Connection in** option is set depending on how you accessed the wizard. The default selection is **Application Resources** if you invoked the wizard from an application, and **Resource Palette** if you invoked the wizard from the Resource Palette. You can change this option at this point.

Figure 64–7 The Specify Producer Name Page (PDK-Java Producer)



3. From the **Target Project** dropdown list, select the project to be configured for the PDK-Java producer connection.

   This should be the same project as the one in which you intend to consume the portlets.

   You can change this option only if you invoked the wizard from the Application Navigator.

4. In the **Producer Registration Name** field, enter a name for the producer registration that is unique among all connections and then click **Next.**

5. In the Specify Connection Details page (Figure 64–8), in the **URL Endpoint** field, enter the producer's URL using the following syntax:

   `http://host:port/context-root/providers`

   Where:

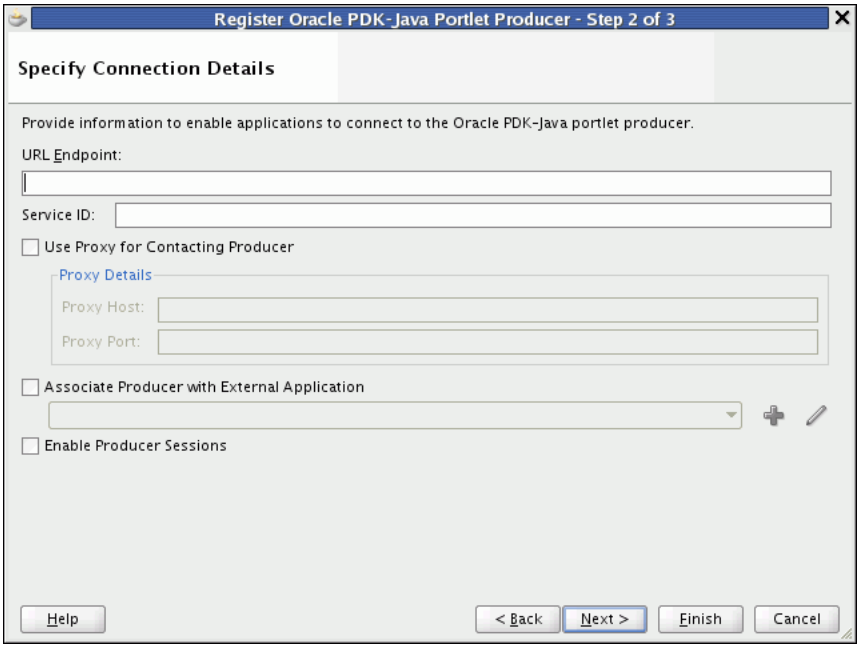
   - *host* is the server to which your producer has been deployed.
   - *port* is the port to which the server is listening for HTTP requests.

- *context-root* is the Web application's context root.
- providers is static text. The text entered here depends on how the producer is deployed.

For example:

```
http://myhost.example.com:7101/myEnterprisePortlets/providers
```

Figure 64–8 The Specify Connection Details Page (PDK-Java Producer)



6. In the **Service ID** field, enter the unique identifier for this producer.

   PDK-Java enables you to deploy multiple producers under a single adapter servlet. The producers are identified by their unique service IDs. A service ID is required only when a service ID or producer name is not appended to the URL endpoint. For example the following URL endpoint requires the service ID, sample:

   ```
   http://myhost:7101/myEnterprisePortlets/providers
   ```

   However, the following URL endpoint, does not require a service ID:

   ```
   http://myhost:7101/myEnterprisePortlets/providers/sample
   ```

7. If the application and the producer are separated by a firewall, an HTTP proxy is needed for communication between the application and the producer. If this is the case for your application, select **Use Proxy for Contacting Producer** and specify the URL and port number of the proxy.

   Note: The proxy fields in this step default to the proxy preferences set in JDeveloper Preferences (from the main menu, choose **Tools > Preferences,** and then select **Web Browser and Proxy.**)

8. Select **Associate Producer with External Application**, then select the application, if this producer must provide authentication to an external application.

For more information, see Section 69.13.3, "Managing External Applications."

This option is available only if you invoked the wizard from the Application Navigator, as external applications are scoped to individual applications.

9. Select **Enable Producer Sessions** to enable sessions between the producer and the consuming application.

   When sessions are enabled, the server maintains session-specific information, such as user name. Message authentication uses sessions, so if the shared key is set, then this option should also be selected.

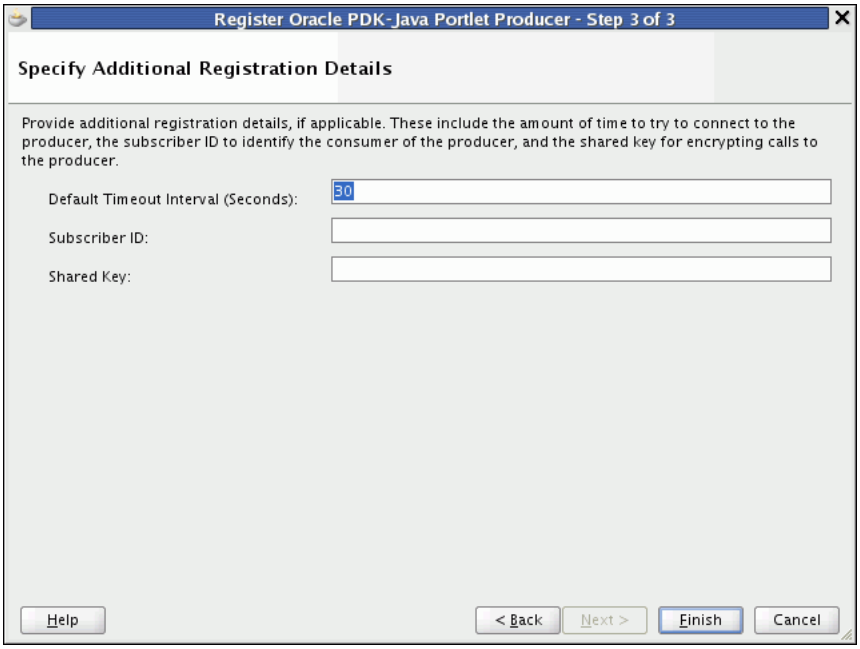
   For sessionless communication between the producer and the server, deselect this option.

10. At this point in the wizard, you can click **Finish** to complete the registration, using the default values for all remaining steps.

    To provide additional details, click **Next** and follow the remaining steps.

11. In the Specify Additional Details page (Figure 64–9), in the **Default Timeout Interval (Seconds)** field, enter the number of seconds to wait for the producer to respond during design time operations.

Figure 64–9 The Specify Additional Registration Details Page (PDK-Java Producer)



12. In the **Subscriber ID** field, enter a string to identify the consumer of the producer being registered.

    When a producer is registered, a call is made to the producer. During the call, the consumer passes the value for Subscriber ID to the producer. This value for Subscriber ID is also passed every time a portlet call is made. If the producer does not see the expected value for Subscriber ID, then it might reject the registration call.

> **Note:** Editing the producer registration to change the Subscriber ID after the initial registration has no effect. To specify a different Subscriber ID, you must reregister the producer.

13. In the **Shared Key** field, enter a shared key to use for producers that are set up to handle encryption and then click **Finish.**

    The shared key is used by the encryption algorithm to generate a message signature for message authentication. Producer registration fails if the producer is set up with a shared key and you enter an incorrect shared key here. The shared key can contain between 10 and 20 alphanumeric characters.

64.2.4 How to Edit Portlet Producer Registration Settings

Both the WSRP and PDK-Java portlet producer registration wizards enable you to access and revise many of the values you entered when you registered the producer.

To edit portlet producer registration settings:

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to edit, and choose **Properties.**
3. Edit the producer registration properties as required, clicking **Next** to step through the pages of the wizard.

   You can also go directly to a specific step by clicking the links in the navigation panel on the left side of the wizard.

   - For information about WSRP Producer settings, see Section 64.2.1, "How to Register a WSRP Portlet Producer."
   - For information about Oracle PDK-Java Producer settings, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

   You cannot edit the **Producer Registration Name.**
4. When you have changed all the necessary settings, click **Finish.**
5. Editing some properties, such as the **WSDL URL** or **URL Endpoint,** requires a producer refresh. When you make such a change, a dialog displays allowing you to refresh the producer immediately, save the changes but do not refresh the producer, or return to the edit producer registration wizard. Click **Yes, No,** or **Cancel** as appropriate.

   > **Note:** While you can edit the value of the **WSDL URL** field, you can point to a different producer only if the new producer has access to the preference store of the old producer, or if the preference store of the old producer has been migrated to that of the new producer.

6. Click **OK** when the producer has been successfully refreshed, if necessary.
7. Once you have completed your edits, consider testing the producer connection to be sure the connection information is valid. For more information, see Section 64.2.5, "How to Test a Portlet Producer Connection."

64.2.5 How to Test a Portlet Producer Connection

The connection testing feature provides a means of testing the validity of a portlet producer connection.

To test a portlet producer connection:

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to test, and choose **Test Producer Connection.**
3. A progress bar appears while the test is underway. A success or failure dialog displays when the test is complete. Click **OK** to close this dialog.

   If the failure dialog displays, consider editing the producer registration details and retesting the producer connection. Additionally, especially if the failure dialog takes a long time to display, ensure that the producer is available. For example, if the producer is provided through the Integrated WLS, ensure that the Integrated WLS is running, and then retest the connection.

64.2.6 How to Refresh a Portlet Producer

When you refresh a portlet producer, the portlets from that producer are also refreshed. Newly added portlets and any updates to existing portlets become available to any applications that are consuming portlets from this producer.

> **Tip:** When a portlet is removed from a producer, be sure to manually delete the portlet from all application pages on which it has been placed. For more information, see Section 64.6, "Deleting Portlets from Application Pages."

To refresh a portlet producer:

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to refresh, and choose **Refresh Producer Registration.**
3. In the Portlet Producer Refresh dialog, click **Yes.**

64.2.7 How to Delete a Portlet Producer

If you no longer want to use a particular producer with your application, you can delete the producer. For information about deleting portlets and relevant page variables, see Section 64.6, "Deleting Portlets from Application Pages."

> **Note:** if you delete a producer, any pages that consume portlets from that producer display an error message that the portlet is unavailable. The portlets continue to be unavailable even if you re-register the producer using the same name.

To delete a portlet producer:

1. Navigate to the producer in the Application Resources panel of the Application Navigator or in the Resource Palette.
2. Right-click the producer to delete, and choose **Delete.**

3. In the Delete Confirmation dialog, click **Yes.**

64.3 Adding Portlets to a Page

Placing a portlet on a Framework application page is a simple matter of dragging the portlet from the Application Resources panel or Resource Palette and dropping it on the page.

This section includes the following subsections:

- Section 64.3.1, "How to Add a Portlet to a Page"
- Section 64.3.2, "What Happens When You Add a Portlet to a Page"
- Section 64.3.3, "What Happens at Runtime"

Before You Begin

Before you can place a portlet on a page, there are a few preparatory steps you must perform before you can take this simple action. These include:

1. Creating a Framework application. For more information, see Section 5.2.1, "How to Create a Framework Application."
2. Creating an application page. For more information, see Section 5.3, "Adding Pages to a Portal."
3. Registering the portlet's producer with the application. For more information, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."
4. Some of the portlets you plan to consume may come from applications that handle their own authentication. In such cases, you must register the application as an external application and identify it to the portlet producer that provides it. For more information, see Chapter 69, "Securing Your WebCenter Portal: Framework Application."
5. Some of the portlets you plan to consume may come from producers that are Secure Sockets Layer (SSL) enabled. When you try to access an SSL-enabled producer, you may be presented with a Security Alert dialog, prompting you to view the producer's certificate and add it to the list of trusted certificates. The Security Alert dialog displays only if the producer uses a security certificate issued by a certificate authority that is not widely accepted. To consume portlets from such a producer, you must first add the producer's security certificate to the keystore. For more information, see Section 69.14, "Registering Custom Certificates with the Keystore."

64.3.1 How to Add a Portlet to a Page

You can add a portlet to a page by dragging and dropping it from the Application Resources panel of the Application Navigator or from the Resource Palette.

To add a portlet to a page:

1. In the Application Navigator, open the application that contains the page (`.jspx` file) to which you want to add the portlet.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

Tip: You can also double-click the page to open it.

4. Under the **Connections** node in the Application Resources panel of the Application Navigator, or in the Resource Palette:
   - If the producer is a WSRP producer, expand the **WSRP Producer** node.
   - If the producer is a PDK-Java producer, expand the **Oracle PDK-Java Producer** node.
5. Expand the node for the portlet producer that contains the portlet to add to the page.

   Under the selected producer, all portlets contained by that producer are listed.
6. Drag the portlet from the producer node directly onto the page.

   You should drag the portlet onto a form on the page. If you do not, a dialog displays prompting you to create a form to contain the portlet. Select:
   - **ADF Faces - Form** if the page contains rich client components. This adds an `af:form` component to the page.
   - **JSF HTML - Form** if the page contains HTML components (for example, if it is an upgraded 10.1.3.2 page). This adds an `h:form` or `tr:form` component to the page, depending on the surrounding document tag.

   Do not select **HTML - Form** as this is not valid for portlets.

Note: You can include any Oracle ADF Faces component or task flow as a child component of a `Show Detail Frame` component. However, portlets contain headers similar to those provided by the `Show Detail Frame` component and can be added to a `Panel Customizable` component directly. There are no additional benefits to including portlets in `Show Detail Frame` components.

Note: If you are adding a PeopleSoft portlet to a page in a Framework application, you must set the `renderPortletInIFrame` portlet tag attribute to `true`.

For more information, see Section 64.4, "Setting Attribute Values for the Portlet Tag."

64.3.2 What Happens When You Add a Portlet to a Page

When you add a portlet to a page, a portlet tag (`adfp:portlet` or `adfph:portlet`) is added to the page source. This is the tag that represents the portlet component. This tag includes attributes that you can edit using the Property Inspector, or in the page source, to further control the behavior and appearance of the portlet. For information about these attributes, see Section 64.4, "Setting Attribute Values for the Portlet Tag."

The type of portlet tag used is determined by the following:

- If the project is configured for rich client components alone, the `adfp:portlet` tag is used.
- If the project is configured for Trinidad components alone, the `adfph:portlet` tag is used.
- If the project is configured for both rich client and Trinidad components, a dialog displays where you can choose which portlet tag to use.

- If the project is not configured for rich client or Trinidad components, the `adfp:portlet` tag is used and the project is automatically configured for rich client components.

This is so that the look and feel of the portlet matches that of other components on the page. For example, if you created your page as described in Section 5.3, "Adding Pages to a Portal," the page is a rich client page. In this case, the portlet is added using the `adfp:portlet` tag.

Example 64–1 A Rich Client Page Containing a Portlet

```
<?xml version='1.0' encoding='US-ASCII'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.0"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:adfp="http://xmlns.oracle.com/adf/faces/portlet">
  <jsp:directive.page contentType="text/html;charset=US-ASCII"/>
  <f:view>
    <af:document>
      <af:form>
        <adfp:portlet value="#{bindings.Portlet11_1}"
                      id="portlet1"
                      renderPortletInIFrame="true"/>
      </af:form>
    </af:document>
  </f:view>
</jsp:root>
```

If you are working with an upgraded 10.1.3.2 application or an application that contains Trinidad components, the application uses HTML components, rather than rich client components. In this case, when you drag a portlet onto a page, the `adfph:portlet` tag is used.

Example 64–2 A Trinidad Page Containing a Portlet

```
<?xml version='1.0' encoding='US-ASCII'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.0"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:tr="http://myfaces.apache.org/trinidad"
          xmlns:adfph="http://xmlns.oracle.com/adf/faces/portlet/html">
  <jsp:directive.page contentType="text/html;charset=US-ASCII"/>
  <f:view>
    <tr:document title="Title 1">
      <tr:form>
        <adfph:portlet value="#{bindings.Portlet12_1}"
                       id="portlet1"/>
      </tr:form>
    </tr:document>
  </f:view>
</jsp:root>
```

If the application page includes one or more Composer components, this may influence where the portlet is placed. For example, in the Structure panel, a portlet placed on a page with a `cust:panelCustomizable` tag, would be placed as illustrated in Example 64–3:

Example 64–3 Hierarchical Placement of the Portlet Tag

```
af:document
  af:form
    cust:panelCustomizable
       adfp:portlet
```

> **Note:** We recommend that you do not mix ADF Faces rich client components with HTML or Trinidad components on the same page. Doing so may produce unexpected results at runtime. Therefore, do not place a rich client portlet inside a Composer HTML component or an HTML portlet inside a Composer rich client component.

For information about Composer tags, see Chapter 20, "Enabling Runtime Editing of Pages Using Composer."

> **Note:** When you drop an instance of OmniPortlet onto your page, open the Property Inspector and ensure that the **AllModesSharedScreen**, under the Display Mode category, is set to `false`, the default value. Setting this property to `true` may prevent you from editing certain sections of your OmniPortlet in the OmniPortlet wizard.

When you add a portlet to a page, a portlet binding is added to the page definition file. This portlet binding includes attributes that specify whether the portlet should automatically listen for parameter changes and events (Example 64–4).

Example 64–4 Portlet Binding

```
<portlet id="LotteryPortlet2_1"
         portletInstance="oracle/adf/portlet/SamplePortlets/ap/Ei7default_03ba18be_012d_1000_8002_
0ae8827e9493"
         class="oracle.adf.model.portlet.binding.PortletBinding"
         retainPortletHeaders="false"
         listenForAutoDeliveredPortletEvents="true"
         listenForAutoDeliveredParameterChanges="true"
         xmlns="http://xmlns.oracle.com/portlet/bindings"/>
```

Setting these attributes to `true` (the default) means that any parameters or events raised that match those supported by the portlet (or that match an alias for one of the supported parameters or events) automatically cause the portlet to be updated accordingly. You can set these attributes to `false` to disable this automatic parameter and event listening off. For example, your page might contain multiple instances of the same portlet that require values to come from different sources. If you disable automatic parameter and event listening, you must manually configure the portlet wiring.

64.3.3 What Happens at Runtime

Once you place a portlet on a page, right-click the page and choose **Run.** This displays the page and runs the portlet in your default browser using JDeveloper's Integrated WLS. Different portlets may require additional runtime configuration. Notably, the content of an OmniPortlet or Web Clipping portlet instance is defined at runtime. For more information about OmniPortlet, see Chapter 65, "Creating Portlets with OmniPortlet." For more information about the Web Clipping portlet, see Chapter 66,

"Creating Content-Based Portlets with Web Clipping." For more information about portlets generally, see Chapter 58, "Overview of Portlets."

When running a portlet that has an Edit mode (in a Framework application, this renders as a Personalize icon (pencil icon) in the portlet header), the Personalize icon displays only to authenticated users (that is, users who have logged in). Anonymous or public users do not see the option to personalize the portlet. Some form of security must be implemented for the portlet-consuming application before users can personalize their view of a portlet. If you are a developer creating portlets and pages, you may want to test your portlet's Edit mode without creating a complete security model for your application. For information about how to add security to enable testing of a portlet's Edit mode, see Section 69.12, "Configuring Basic Authentication for Testing Portlet Personalization."

Note: To be able to add portlets to your page at runtime, you must add at least one portlet to that page at design time. Adding a portlet at design time ensures that the following is added to the `<definitionFactories>` element of the `DataBindings.cpx` file:

```
<factory nameSpace="http://xmlns.oracle.com/portlet/bindings"
className="oracle.adf.model.portlet.binding.PortletBindingDefFactor
yImpl"/>
<dtfactory
className="oracle.adfdtinternal.view.faces.portlet.PortletDefinitio
nDTFactory"/>
```

This entry is required to enable consumption of portlets at runtime.

If a portlet supports parameters or events and the automatic parameter and event listening is enabled, any changes to the supported parameters and events (or to parameters and events that are aliased) automatically update the portlet.

When running a portlet from a producer associated with an external application, a link to update login information is displayed. Clicking the link displays a credential provisioning page for entering external application credentials. After specifying valid credentials the portlet displays content appropriately. For more information about external applications, see Section 69.13, "Working with External Applications."

64.4 Setting Attribute Values for the Portlet Tag

In the source code view of a page, each portlet is represented by an `adfp:portlet` tag (or `adfph:portlet` tag), which includes a set of required and optional attributes. Required attributes, `value` and `portletType`, are provided automatically by the framework, and must not be altered. Optional attribute values are relevant when support for the attribute is built into the portlet. For example, you can set `isAboutModeAvailable` to `true`, but if no About mode has been defined for the portlet, then the attribute setting does not affect the portlet.

Portlets also support a set of style-related attributes, which are discussed more fully in Section 22.10, "Applying Styles to Components."

The portlet tag uses many attributes, which you can set at design time either through the JDeveloper Property Inspector or in the source code as attributes of the tag.

This section includes the following subsections:

- Section 64.4.1, "How to Set Attribute Values for the Portlet Tag Using the Property Inspector"
- Section 64.4.2, "How to Set Attribute Values for the Portlet Tag in Source Code"
- Section 64.4.3, "Common Attributes of the Portlet Tag"
- Section 64.4.4, "Appearance Attributes of the Portlet Tag"
- Section 64.4.5, "Behavior Attributes of the Portlet Tag"
- Section 64.4.6, "Portlet Modes Attributes of the Portlet Tag"
- Section 64.4.7, "Style Attributes of the Portlet Tag"
- Section 64.4.8, "Binding Attributes of the Portlet Tag"
- Section 64.4.9, "Customization Attributes of the Portlet Tag"
- Section 64.4.10, "Other Attributes of the Portlet Tag"
- Section 64.4.11, "What You May Need to Know About Maximize, Minimize, Restore, and Move"
- Section 64.4.12, "What You May Need to Know About IFRAMEs"

64.4.1 How to Set Attribute Values for the Portlet Tag Using the Property Inspector

The Property Inspector provides a quick and easy way to set attribute values for the portlet tag without having to edit the source code yourself.

To set attribute values for the portlet tag using the Property Inspector:

1. In the Application Navigator, open the application that contains the page on which the portlet appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

   Tip: You can also double-click the page to open it.

4. In the design view, select the portlet whose attributes you want to set.
5. In the Property Inspector, click the appropriate tab and set the desired attribute.

   Repeat this step as often as required.
6. Save your changes.
7. To test the changes you have made, right-click the page and choose **Run.**

64.4.2 How to Set Attribute Values for the Portlet Tag in Source Code

If you prefer working in source code, you can set attribute values for the portlet tag directly there.

To set attribute values for the portlet tag in source code:

1. In the Application Navigator, open the application that contains the page on which the portlet appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

   Tip: You can also double-click the page to open it.

4. In the design view, select the portlet whose attributes you want to set.
5. Click the **Source** tab. The portlet that you selected is highlighted in the source code.
6. Make your changes directly to the source code. Example 64–5 shows an edited portlet tag.

Example 64–5 An Edited Portlet Tag

```
<adfp:portlet value="#{bindings.portlet1}"
              portletTypes="/oracle/adf/portlet/WsrpPortletProducer1/applicationPortlets/
                            E0default_b452f828_010a_1000_8002_82235f57eaa8"
              allModesSharedScreen="true"
              isMaximizable="true"
              isMinimizable="true"/>
```

7. Save your changes.
8. To test the changes you have made, right-click the page and choose **Run.**

64.4.3 Common Attributes of the Portlet Tag

Table 64–1 describes the common attributes of the portlet tag.

Table 64–1 Common Attributes of the Portlet Tag

Attribute	Value	Description
`id`	Text string. For example: `id="newsBrief"` The value must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Z a-z) or an underscore (_). ■ Subsequent characters must be ASCII letter or digits (a-Z a-z 0-9), underscores (_), or dashes (-).	The unique identifier of the portlet. This attribute is populated with a unique value by default when you add the portlet to a page.
`title`	Text string. For example: `title="Announcements"`	The portlet title, which is displayed in the portlet header. The value specified here takes precedence over any title specified elsewhere (for example, in the portlet markup). If no value is specified here, the portlet extracts its title from the portlet markup (response). If no value is specified either here or in the portlet markup, the portlet extracts its title from the portlet definition. **Note:** Supplying a value to the `title` attribute at design time means that any change made to the title at runtime in Edit or Edit Defaults mode is ignored.

`width`	Number expressed in pixels or as a percentage of available area:
	■ For pixels, enter *n*px, for example:
	`width = 300px`
	■ For percentage, enter *n*%, for example:
	`width = 50%`
		The width of the area to allow for portlet display.
		If the actual portlet width is larger than the `width` value entered here, a scrollbar appears, provided `displayScrollBar` is set to `auto` or `true`. If `displayScrollBar` is set to `false`, and the actual portlet width exceeds the value expressed for the `width` attribute, the `width` attribute value is considered and the portlet content is truncated.

Table 64–1 (Cont.) Common Attributes of the Portlet Tag

Attribute	Value	Description
`height`	Number expressed in pixels, for example: `height = 300px`	The height of the area to allow for portlet display. If the actual portlet height is larger than the `height` value entered here, a scrollbar appears, provided `displayScrollBar` is set to `auto` or `true`. If `displayScrollBar` is set to `false`, and the actual portlet height exceeds the value expressed for the `height` attribute, the `height` attribute value is considered and the portlet content is truncated.
`icon`	URI to an image. For example: `icon="coffee.png"` In the Property Inspector, click the Property Menu icon next to the field and then choose **Edit** to locate and select the required image. The value must be an absolute URI or a URI that is resolvable relative to the current page or the application context root. The URI provided in the preceding example is stored at the application context root, therefore a full path is not required.	A URI specifying the location of an image to use as an icon, displayed to the left of the portlet title in the portlet header. You can use the icon to indicate the portlet's purpose, to reinforce branding, as a content indicator, or for some other reason.
`partialTriggers`	One or more component IDs. For example: `partialTriggers="_id1 _id2 componentID5"` Separate component IDs with spaces.	The IDs of the components that trigger a partial update. The portlet listens on the specified trigger components. If a trigger component receives a trigger event that causes it to update in some way, this portlet also requests to be updated.

64.4.4 Appearance Attributes of the Portlet Tag

Table 64–2 describes the appearance attributes of the portlet tag.

Table 64–2 Appearance Attributes of the Portlet Tag

Attribute	Value	Description

`expansionMode`	`minimized`
	`normal`
	Default: `normal`
		The default state of the portlet:
		▪ `minimized`—The portlet's default display mode is collapsed (minimized).
		▪ `normal`—The portlet's default display made is neither collapsed nor expanded to the width of the page.

`allModesSharedScreen`	`auto`
	`false`
	`true`
	Default: `auto`
		Whether a change in portlet mode renders the new mode on a new page, rather than the page on which the portlet resides.
		▪ `auto`—All portlet modes are displayed inline if the remote portlet is configured, through the `com.oracle.portlet.requireIFrame` container runtime option in its `portlet.xml` file, to require an inline frame (IFRAME). This ensures that Oracle Bridge portlets are displayed inline.
		▪ `false`—All portlet modes, except View (JSR 286) or Show (PDK-Java), are rendered each on their own page. This is useful for portlets such as OmniPortlet and the Web Clipping portlet, which require that modes other than Show mode display on pages other than the page on which the portlet resides.
		▪ `true`—All portlet modes are displayed inline. One mode is swapped out for another on the same page. In other words, this attribute enables all portlet modes to display without leaving the context of a given page.

`renderPortletInIFrame`	`auto`
	`false`
	`true`
	Default: `auto`
		Whether the portlet is rendered in an IFRAME:
		▪ `auto`—Whenever possible, the portlet consumer attempts to rewrite the portlet markup so that it renders correctly in an ADF page. Otherwise, the portlet is rendered in an IFRAME.
		▪ `false`—The portlet is never rendered in an IFRAME.
		▪ `true`—The portlet is always rendered in an IFRAME.
		For more information, see Section 64.4.12, "What You May Need to Know About IFRAMEs."

`displayScrollBar`	`auto`
	`false`
	`true`
	Default: `auto`
		Whether a scroll bar is displayed:
		▪ `auto`—Render a scroll bar if the portlet content does not fit the `width` and `height` specified.
		▪ `false`—Never render a scroll bar. If the portlet content does not fit the `height` and `width` specified, the portlet renders in its actual size.
		▪ `true`—Always render a scroll bar.

Table 64–2 (Cont.) Appearance Attributes of the Portlet Tag

Attribute	Value	Description
displayHeader	false true Default: true	Whether the portlet header is displayed: ■ false—The portlet header is not displayed. Icons and links normally displayed in the header are hidden. If isSeededInteractionAvailable is set to true, users can access portlet menus and icons by rolling the mouse over the portlet. A fade-in/fade-out toolbar appears, from which users can select Actions menu options. ■ true—The portlet header is displayed. Consequently, header-based icons and links are displayed.
displayShadow	false true Default: true	Whether to display a shadow decoration around the portlet: ■ false—Do not display a shadow decoration. ■ true—Display a shadow decoration.
rendered	false true Default: true	Whether the portlet is rendered. ■ false—Do not render the portlet. No output is rendered. ■ true—Render the portlet. This is the recommended setting. Setting this attribute to false causes problems when you run the page.
background	dark light medium Default: medium	The style selector to apply to the skin used by the portlet: ■ dark—Apply the dark style selector to the skin. ■ light—Apply the light style selector to the skin. ■ medium—Apply the medium style selector to the skin. This provides a way for you to apply a different look and feel to each portlet on an page.
shortDesc	Text string. For example: shortDesc="Portlet for entering display text in place."	A short description of the portlet.
displayActions	always onHover Default: always	Whether seeded interactions for the portlet are shown: ■ always—Always show seeded interactions. ■ onHover—Show seeded interactions when users move the mouse over the portlet.

Table 64–2 (Cont.) Appearance Attributes of the Portlet Tag

Attribute	Value	Description
showMoveAction	menu none Default: menu	Whether to display the **Move** command in the portlet's **Action** menu: ■ menu—Display the **Move** command on the portlet's **Action** menu. ■ none—Do not display the **Move** command. There is a difference in the way that the **Move** command behaves at design time and at runtime. For more information, see Section 64.4.11, "What You May Need to Know About Maximize, Minimize, Restore, and Move."
showRemoveAction	menu none Default: menu	Whether to display the Remove icon on the portlet chrome: ■ menu—Display the **Remove** command on the portlet's **Action** menu. ■ none—Do not display the Remove icon. There is a difference in the way that the Remove icon behaves at design time and at runtime. For more information, see Section 64.4.11, "What You May Need to Know About Maximize, Minimize, Restore, and Move." **Note:** This attribute is available only for the adfp:portlet tag and not for the adfph:portlet tag.
showResizer	always never Default: always	Whether to display the resize handle at the bottom right corner of the portlet. ■ always—Always display the resize handle. ■ never—Never display the resize handle. **Note:** This attribute is available only for the adfp:portlet tag and not for the adfph:portlet tag.
showMinimizeAction	chrome none Default: chrome	Whether to display the Minimize icon on the portlet chrome: ■ chrome—Display the Minimize icon on the portlet chrome. ■ none—Do not display the Minimize icon. There is a difference in the way that the Minimize icon behaves at design time and at runtime. For more information, see Section 64.4.11, "What You May Need to Know About Maximize, Minimize, Restore, and Move."

64.4.5 Behavior Attributes of the Portlet Tag

Table 64–3 describes the behavior attributes of the portlet tag.

Table 64–3 Behavior Attributes of Portlet Tag

Attribute	Value	Description
`partialTriggers`	One or more component IDs. For example: `partialTriggers="_id1 _id2 componentID5"` Separate component IDs with spaces.	The IDs of the components that trigger a partial update. The portlet listens on the specified trigger components. If a trigger component receives a trigger event that causes it to update in some way, this portlet also requests to be updated.

`submitUrlParamters`	`false`
	`true`
	Default: `false`
		Whether parameters in portlet links that point to the page on which the portlet is placed are made available to the page:
		▪ `false`—Parameters are not made available to the page. Rather, they are available only inside the portlet initiating the request.
		▪ `true`—Parameters available on the container page.

64.4.6 Portlet Modes Attributes of the Portlet Tag

Portlet Modes attributes control the rendering of mode-switching UI actions, such as entering edit mode. The ability to render a portlet in a particular mode depends on the modes supported by the portlet and the user authorization. For example, if the `isCustomizeModeAvailable` attribute is set to `true`, but the action is not supported in the portlet, then the attribute setting does not affect the portlet.

Portlet Modes attributes, described in Table 64–4, are value binding expressions that evaluate to `true` or `false`:

- `true` means the portlet is allowed to render in the named mode.
- `false` means the portlet is not allowed to render in the named mode.

Table 64–4 Portlet Modes Attributes of the Portlet Tag

Attribute	Value	Description

`isAboutModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render an **About** command on the portlet's **Actions** menu.
		Users choose **About** to invoke the portlet's About mode.

`isConfigModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Configure** command on a JSR 286 portlet's **Actions** menu.
		Users choose **Configure** to open the portlet's Configuration settings.

`isCustomizeModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Customize** icon in the portlet header.
		Site administrators choose **Customize** to edit a portlet's default personalization data.

`isDetailModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Details** command on a PDK-Java portlet's **Actions** menu.
		Users choose **Details** to open the portlet in Full Screen mode.

Table 64–4 (Cont.) Portlet Modes Attributes of the Portlet Tag

Attribute	Value	Description

`isHelpModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Help** command on the portlet's **Actions** menu.
		Users choose **Help** to open the portlet's Help page.

`isPrintModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Print** command on a JSR 286 portlet's **Actions** menu.
		Users choose **Print** to displays a printer-friendly version of the portlet.

`isNormalModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Refresh** command on the portlet's **Actions** menu.
		Users choose **Refresh** to redraw the portlet independent of any other content on the page (also known as a partial-page refresh).

`isPersonalizeModeAvailable`	`false`
	`true`
	Default: `true`
		Whether to render a **Personalize** icon in the portlet header.
		Users choose **Personalize** to alter their personal view of the portlet. This mode is equivalent to the Edit mode selection in the Standards-based Java Portlet (JSR286) Wizard.
		The **Personalize** icon displays only to authenticated users (that is, users who are logged in). It does not display to public or unauthenticated users. You must implement some form of application security for users to be able to personalize their portlet views.
		If you are a developer creating portlets, and you want to test the Personalize mode without creating a complete security model for your application, then see Section 69.12, "Configuring Basic Authentication for Testing Portlet Personalization."
		Note: A typical personalization setting is Portlet Title. You can set Portlet Title at design time, by providing a value for the `title` attribute. Consider however that supplying a value to the `title` attribute at design time prevents personalization and customization of the portlet title at runtime.

`isPreviewModeAvailable`	`false`
	`true`
	Default: `false`
		Whether to enable previewing of portlet content.
		This mode has no particular application in Framework applications, but it is used in Oracle Portal's Portlet Repository, where it renders as a magnifying glass icon, which users click to preview a portlet.

64.4.7 Style Attributes of the Portlet Tag

Table 64–5 describes the style attributes of the portlet tag.

Table 64–5 Style Attributes of the Portlet Tag

Attribute	Value	Description
`contentStyle`	One or more CSS styles. These should be in compliance with, at least, CSS 2.0 and take the following format: `contentStyle="color:rgb(255,0,255); font-family:Arial Helvetica Geneva sans-serif;font-size:large;"`	The CSS style to apply to the portlet content. Values entered here take precedence over styles specified in the `inlineStyle` attribute and those included in a CSS or skin on the specific portlet instance. For more information, see Understanding contentStyle and inlineStyle Properties.
`inlineStyle`	One or more CSS styles. These should be in compliance with, at least, CSS 2.0 and take the following format: `inlineStyle="color:rgb(255,0,255); font-family:Arial Helvetica Geneva sans-serif;font-size:large;"`	The CSS style to apply to the whole portlet. Values entered here take precedence over styles included in a CSS or skin on the specific portlet instance. For more information, see Understanding contentStyle and inlineStyle Properties.

64.4.8 Binding Attributes of the Portlet Tag

Table 64–6 describes the binding attributes of the portlet tag.

Table 64–6 Binding Attributes of the Portlet Tag

Attribute	Value	Description
`binding`	Name of a managed bean. For example: `binding="#{frameActionsBean.Binding}"` In the Property Inspector, click the Property Menu icon next to the field and then choose **Edit** to select a managed bean and specify the relevant managed bean property.	A binding reference to store the component instance. The binding reference binds an instance of the portlet to a managed bean property. Managed beans are any JavaBeans used by the application that are registered in the JSF `faces-config.xml` file.

64.4.9 Customization Attributes of the Portlet Tag

Table 64–7 describes the customization attributes of the portlet tag.

Table 64–7 Customization Attributes of the Portlet Tag

Attribute	Value	Description

`customizationAllowed`	`false`
	`true`
	Default: `true`
		Whether design time customizations of the portlet tag are allowed on this portlet.

`customizationAllowedBy`	Text string	The roles for which design time customizations are allowed. This attribute enables you to allow customizations, but restrict who can actually perform them.

64.4.10 Other Attributes of the Portlet Tag

Table 64–8 describes the other attributes of the portlet tag.

Table 64–8 Other Attributes of the Portlet Tag

Attribute	Value	Description
iframeDtd	loose none strict Default: loose	Which DTD, if any, is specified in the doctype declaration that is created when portlet content is rendered inside a IFRAME: ■ none—No DTD is specified. This relaxes the restrictions on the HTML content being technically conformant HTML. Browsers usually handle such HTML acceptably, however, because some CSS style sheet from the ADF Faces page consuming the portlet is also imported into the IFRAME document, for that style sheet to work correctly, it may be necessary to declare the content conformant to the loose or strict DTDs. ■ loose—The DTD http://www.w3.org/TR/html/loose.dtd is used. ■ strict—The DTD http://www.w3.org/TR/html/strict.dtd is used

64.4.11 What You May Need to Know About Maximize, Minimize, Restore, and Move

To accommodate the needs of the development environment, the behavior of the actions Minimize, Maximize, Restore, and Move for `Show Detail Frame` and `portlet` components differs between design time and runtime. At design time, these actions persist in a given WLS session, but do not persist over sessions (*session* means the time between starting and stopping WLS). At runtime, these actions persist both during a given WLS session and across sessions.

This difference has been introduced to enable an automatic reset of an application page at design time.

If persisting across sessions is not required at runtime, then a simple modification to the application's `web.xml` file can turn it off. Go to the following parameter setting in the application's `web.xml` file (Example 64–6):

Example 64–6 Persistence Setting in the Application's web.xml File

```
<context-param>
   <param-name>oracle.adf.view.faces.CHANGE_PERSISTENCE</param-name>
   <param-value>oracle.adfinternal.view.faces.change.HybridChangeManager</param-va
lue>
</context-param>
```

Replace it with the following (Example 64–7):

Example 64–7 Turning Runtime Persistence Off in the Application's web.xml File

```
<context-param>
   <param-name>oracle.adf.view.faces.CHANGE_PERSISTENCE</param-name>
   <param-value>oracle.adf.view.faces.change.SessionChangeManager</param-value>
</context-param>
```

If security has been implemented on the application, then the Minimize, Maximize, Restore, and Move actions display only to users with Customize privileges. They do

not display to users with Personalize privileges. Customize users can test the effect of these actions by following these steps at design time:

1. Run the application page using JDeveloper's Integrated WLS.
2. Log in as the administrator.
3. Maximize a portlet. Move portlets around. Make whatever changes you want using the relevant actions commands.
4. Log out, then log in as a user and check the effects of your actions.

64.4.12 What You May Need to Know About IFRAMEs

Rendering a portlet directly on a page provides a better user experience as compared to placing it in an inline frame, or IFRAME. However, at times, it may be required to include the portlet markup inside an IFRAME. You can use the `renderPortletInIFrame` attribute to determine whether or not a portlet should be rendered in an IFRAME.

The default setting of the `renderPortletInIFrame` attribute is `auto`. This causes the portlet consumer to attempt to rewrite the portlet markup whenever possible, so that it renders correctly within the Oracle ADF page without the need for an IFRAME.

However, in the following circumstances, the portlet is rendered within an IFRAME:

- The parser throws an exception as it is not able to parse the markup.
- The portlet code contains a multi-part form post.
- The portlet code contains a file upload element.
- The portlet developer explicitly requested that the portlet be rendered in an IFRAME by setting the `com.oracle.portlet.requireIFrame` container runtime option in the `portlet.xml` file to `true`.

> **Note:** Portlets created using the Oracle JSF Portlet Bridge have the `requireIFrame` container runtime option set to `true` as these portlets are too complex to render directly on Oracle ADF pages due to JavaScript issues.

In some circumstances, the portlet consumer may be unable to rewrite the portlet markup and JavaScript to accommodate the portlet in the Oracle ADF page. If this is the case, you may find that the portlet does not behave as expected, for example, buttons do nothing or you get JavaScript errors in the console. If this happens, you should set the `renderPortletInIFrame` attribute to `true` so that the portlet is always rendered in an IFRAME.

Some examples of when you should set `renderPortletInIFrame` to `true` are when:

- The portlet code builds up element names dynamically in JavaScript.
- The portlet code uses multiple forms and references them in JavaScript by index.
- JavaScript library code references elements within the portlet.

> **Note:** If you render a portlet within an IFRAME, then manipulating `window.location` may give unexpected results. If your portlet uses `window.location`, then you should ensure that your JavaScript is robust enough to handle the case where the portlet renders itself inside an IFRAME.

If you want to ensure that a portlet is never rendered in an IFRAME, for example for accessibility reasons, set the `renderPortletInIFrame` attribute to `false`. Note however, that HTML markup from a portlet that is not rendered in an IFRAME may interfere with other components on the Oracle ADF page.

For more information about the accessibility implications of using IFRAMEs, see Section 2.9, "Overview of WebCenter Accessibility Features."

64.5 Copying Portlets

When you copy portlets, the portlets and their copies must reside within the same application. For example, you can copy a portlet from one page in an application to another page in the same project in that application, or from one place on a page to another place on the same page. You can also copy a portlet from one project to another project in the same application, if the target project is configured for consuming portlets. The copies are references to the same portlet instance, so customizations or personalizations made to any instance of the portlet (original or copy) affect all the other instances.

Copying a portlet is more than a matter of copying and pasting the portlet view tag. It involves copying portlet-related entries from the application page's source. It may also involve copying portlet-related entries from the page definition file and removing duplicate portlet binding information or creating a new method in the copied portlet's binding bean.

When a portlet is copied, the target page must be an Oracle ADF Faces page. Any preexisting code on the target page must reflect that. This is quite easy to accomplish. When JDeveloper creates a new JSF page, it contains pure JSF tags. The first time you drop an Oracle ADF Faces component onto the page, tags are automatically updated to be Oracle ADF Faces tags. For example, an `<html>` tag becomes `<afh:html>`, `<head>` and `<title="title">` tags become `<afh:head title="title">`, and so on. Therefore, a simple way to ensure the conversion of the target page to an Oracle ADF Faces page is to place any Oracle ADF Faces component on the target page. This performs any required code conversion for you automatically.

This section includes the following subsections:

- Section 64.5.1, "How to Copy a Portlet and Place it on the Same Page"
- Section 64.5.2, "How to Copy a Portlet from One Application Page to Another"

64.5.1 How to Copy a Portlet and Place it on the Same Page

Because all of the page's resources are available to both portlet instances when you copy a portlet to the same page, there is no requirement to copy portlet-related information from the page's Page Definition file. It is just a matter of copying and pasting the portlet's view tag, and assigning a unique identifier to the copy.

To copy and place a portlet on the same page:

1. In the Application Navigator, open the application that contains the page on which the portlet to copy appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

   Tip: You can also double-click the page to open it.

4. In the design view, select the portlet to copy.
5. Click the **Source** tab. The portlet that you selected is highlighted in the source code.
6. Copy the portlet tag (Example 64–8).

Example 64–8 Code Fragment to be Copied When Copying a Portlet

```
<f:view>
   <afh:html binding="#{backing_portlet_page.html1}" id="html1">
      <afh:head title="portlet_page" binding="#{backing_portlet_page.head1}"
         id="head1">
         <meta http-equiv="Content-Type"
            content="text/html;charset=windows-1252"/>
      </afh:head>
      <afh:body binding="#{backing_portlet_page.body1}" id="body1">
         <h:form binding="#{backing_portlet_page.form1}" id="form1">
            <adfp:portlet value="#{bindings.portlet1}"
               portletType="/oracle/adf/portlet/
               pdksampleproducer_1153245807295/applicationPortlets/
               Portlet2_82d49b79_010c_1000_8006_82235ffc4e2b"
               binding="#{backing_portlet_page.portlet1}"
               id="portlet1"
               isCustomModesAvailable="true"/>
         </h:form>
      </afh:body>
   </afh:html>
</f:view>
```

7. Paste the copied code fragment into the desired location of the page source.
8. Provide a unique value for the copy's `id` attribute (Example 64–9).

Example 64–9 Changing the Portlet ID

```
<adfp:portlet value="#{bindings.portlet1}"
   portletType="/oracle/adf/portlet/
   pdksampleproducer_1153245807295/applicationPortlets/
   Portlet2_82d49b79_010c_1000_8006_82235ffc4e2b"
   binding="#{backing_portlet_page.portlet1}"
   id="portlet2"
   isCustomModesAvailable="true"/>
```

Note: On a given page, each portlet must have a unique ID.

9. In the page source, if the copied portlet's `adfp:portlet` tag has a binding attribute, for example:

   ```
   binding="#{backing_untitled2.portlet1}"
   ```

Then either remove this binding, or create a new method in the binding bean by opening the managed bean class for this managed bean and defining the new method.

For example, if portlet1 is copied, the pasted copy becomes portet2 in the managed bean class, as shown in Example 64–10.

Example 64–10 Creating a New Method for a Managed Bean in the Managed Bean Class

```
.
private PortletBase portlet2;
public void setPortlet2(PortletBase portet2)  {
     this.portlet2 = portlet2;
}
.
public PortletBase getPortlet2()  {
     return portlet2;
}
```

10. From the main menu, choose **File > Save All.**

64.5.2 How to Copy a Portlet from One Application Page to Another

When you copy a portlet from one page to another in an application, portlet-related code must also be copied from the source page's Page Definition file. This section describes the steps related to both copying from one application page to another and from one application project to another.

To copy a portlet from one page to another:

1. In the Application Navigator, open the application that contains the page on which the portlet to copy appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

   Tip: You can also double-click the page to open it.

4. In the design view, select the portlet to copy.
5. Click the **Source** tab. The portlet that you selected is highlighted in the source code.
6. Copy the portlet tag (Example 64–11).

Example 64–11 Source Page Code Fragment to Be Copied When Copying a Portlet

```
<f:view>
   <afh:html binding="#{backing_portlet_page.html1}" id="html1">
      <afh:head title="portlet_page" binding="#{backing_portlet_page.head1}"
         id="head1">
         <meta http-equiv="Content-Type"
            content="text/html;charset=windows-1252"/>
      </afh:head>
      <afh:body binding="#{backing_portlet_page.body1}" id="body1">
         <h:form binding="#{backing_portlet_page.form1}" id="form1">
            <adfp:portlet value="#{bindings.portlet1}"
               portletType="/oracle/adf/portlet/
               pdksampleproducer_1153245807295/applicationPortlets/
               Portlet2_82d49b79_010c_1000_8006_82235ffc4e2b"
```

```
                binding="#{backing_portlet_page.portlet1}"
                id="portlet1"
                isCustomModesAvailable="true"/>
          </h:form>
       </afh:body>
    </afh:html>
</f:view>
```

7. Go to the application page to which to copy the portlet (the target page).

   Portlets can reside only on Oracle ADF Faces pages. If the target page does not contain Oracle ADF Faces components, then ensure that the container objects—that is, any tags the portlet tag is nested in—use Oracle ADF tags.

   If you are copying the portlet to a page in a different project, the target project must be configured for consuming portlets. To configure the project, you must register a portlet producer with the project. For information, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."

8. Click the **Source** tab and paste the copied code fragment into the desired location of the page source.

9. In the Application Navigator, right-click the source page (the page from which the portlet was copied), and choose **Go to Page Definition.**

10. Click the **Source** tab and copy the portlet binding from the source page's page definition file (Example 64–12).

Example 64–12 Code Fragment to Be Copied From a Page Definition File

```
<portlet id="portlet1"
  portletInstance="/oracle/adf/portlet/pdksampleproducer_
1153245/applicationPortlets/Portlet2_82d49_010c_1000_8006_82235"
  class="oracle.adf.model.portlet.binding.PortletBinding"
  xmlns="http://xmlns.oracle.com/portlet/bindings"/>
```

> **Note:** When the portlet being copied includes parameters, be sure to include the copied portlet's portlet parameters and the page variables linked to the portlet parameters in the copy.

11. In the Application Navigator, right-click the target page and choose **Go to Page Definition.**

    If a page definition does not exist for the page, you are asked if you want to create one. Click **Yes.**

12. Click the **Source** tab and paste the portlet binding you copied from the source (along with relevant portlet parameters and the page variables associated with those parameters).

    Paste the code between the `<executables>` tags. You may need to add a closing `</executables>` tag and ensure that the opening tag does not contain a slash (`/`).

13. From the main menu, choose **File > Save All.**

64.6 Deleting Portlets from Application Pages

When you delete a portlet from an application page, if the portlet had parameters, then you should also delete page variables associated with those parameters from the application page's Page Definition file.

To delete a portlet and its related page variables from a page:

1. In the Application Navigator, open the application that contains the page on which the portlet to delete appears.
2. Expand the project that contains the page.
3. Locate the page, right-click it, and then choose **Open.**

   Tip: You can also double-click the page to open it.

4. In the design view, right-click the portlet to delete and choose Delete.

   This deletes the portlet from the page and the portlet binding from the Page Definition file.
5. If the portlet included variables, in the Application Navigator, right-click the page and choose **Go to Page Definition.**
6. Click the **Source** tab and locate the page variables associated with the deleted portlet, and delete them from the page definition file.

   For example, if you deleted portlet1, you would also delete the highlighted variables in Example 64–13:

Example 64–13 Deleting Portlet-Related Page Variables from a Page Definition File

```
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
      version="10.1.3.38.97" id="untitled1PageDef"
      Package="project1.pageDefs">
   <parameters/>
   <executables>
      <variableIterator id="variables">
            <variable Name="portlet1_param1" Type="java.lang.Object"/>
            <variable Name="portlet1_param2" Type="java.lang.Object"/>
            <variable Name="portlet2_param1" Type="java.lang.Object"/>
            <variable Name="portlet2_param2" Type="java.lang.Object"/>
      </variableIterator>
      <portlet id="portlet2" portletInstance="/oracle/adf/portlet/
            PdkPortletProducer2_1154100666247/applicationPortlets/
            Portlet1_b5e49696_010c_1000_8008_8c5707ef9c4f"
            class="oracle.adf.model.portlet.binding.PortletBinding"
            xmlns="http://xmlns.oracle.com/portlet/bindings">
         <parameters>
              <parameter name="param1" pageVariable="portlet2_param1"/>
              <parameter name="param2" pageVariable="portlet2_param2"/>
         </parameters>
      </portlet>
   </executables>
   <bindings/>
</pageDefinition>
```

7. From the main menu, choose, select **File > Save All**.

64.7 Consuming WebCenter Services Portlets

WebCenter Services Portlets is a preconfigured, out-of-the-box producer that enables you to expose WebCenter Portal service task flows in other applications as WSRP portlets or pagelets.

WebCenter Services Portlets provides the following portlets:

- Document Manager—Displays folders, files, and wikis from the WebCenter Content repository
- Blogs—Displays blog posts from a selected location in the WebCenter Content repository
- Discussion Forums—Displays all discussions and their respective replies and enables users to perform various operations based on their privileges
- Announcements—Displays all current announcements and enables users to perform various operations based on their privileges
- Lists—Displays user-created lists and provides controls for creating lists and adding list data
- Polls Manager—Enables users to perform administrative operations on polls
- Take Polls—Displays the most recently published available poll, or a specific poll identified by the pollId parameter
- Worklist—Enables users to view and take action on all tasks and notifications from one or more Oracle BPEL Server
- Mail—Displays a mail inbox
- Activity Stream—Provides an overview of the most recent activities performed by a user's connections
- Tag Cloud—Displays a tag cloud, which is a visual depiction of all the tags used on the page

You can consume WebCenter Services Portlets in the following applications:

- Oracle Portal
- Oracle WebLogic Portal
- Oracle WebCenter Interaction, using Oracle WebCenter Portal's Pagelet Producer

The WebCenter Services Portlets producer is a WSRP producer. As such, you register it with your application in the same way as you would any other WSRP producer.

Note: The WebCenter Services Portlets producer is a secured producer, so when you register the producer, you must use the same token as was used when configuring the producer in WebCenter Portal.

For information about configuring the WebCenter Services Portlets producer, see the "Configuring WebCenter Services Portlets" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following subsections:

- Section 64.7.1, "Consuming WebCenter Services Portlets in Oracle Portal"

- Section 64.7.2, "Consuming WebCenter Services Portlets in Oracle WebLogic Portal"
- Section 64.7.3, "Consuming WebCenter Services Portlets in Oracle WebCenter Interaction"
- Section 64.7.4, "Setting WebCenter Services Portlets Parameters"

64.7.1 Consuming WebCenter Services Portlets in Oracle Portal

To consume WebCenter Services Portlets in Oracle Portal:

1. Register the WebCenter Services Portlets WSRP producer in Oracle Portal.

   For information about how to do this, see the "Securing Access to Web Services Remote Portlets" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle Portal*.

2. After registering the producer, you can then add any of the portlets provided by the producer to pages in your Oracle Portal application.

   For information about how to do this, see the "Adding a Portlet to a Page" section in the *Oracle Fusion Middleware User's Guide for Oracle Portal*.

64.7.2 Consuming WebCenter Services Portlets in Oracle WebLogic Portal

To consume WebCenter Services Portlets in Oracle WebLogic Portal:

1. Register the WebCenter Services Portlets WSRP producer in Oracle WebLogic Portal.

   For information about how to do this, see the "Adding a Producer" section in the *Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal*.

2. Add the remote portlets to the Portal Library.

   For information about how to do this, see the "Adding a Remote Portlet to the Portal Library" section in the *Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal*.

3. The WebCenter Services Portlets producer is a secured WSRP producer, so you must set up the SAML security.

   For information about how to do this, see the "SAML Security Between a WebLogic Portal Consumer and a WebCenter Framework Producer" section in the *Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal*.

4. After registering the producer and setting up the security, you can then add any of the portlets provided by the producer to pages in your Oracle WebLogic Portal application.

   For information about how to do this, see the "Adding Contents to a Page" section in the *Oracle Fusion Middleware Portal Development Guide for Oracle WebLogic Portal*.

64.7.3 Consuming WebCenter Services Portlets in Oracle WebCenter Interaction

To consume WebCenter Services Portlets in Oracle WebCenter Interaction:

1. To make WebCenter Services Portlets available in Oracle WebCenter Interaction, you must first register the producer as a pagelet producer using Oracle WebCenter Portal's Pagelet Producer.

For information about how to do this, see the "Registering WSRP Portlet Producers" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

Note: You must select the same **Token Profile** that was specified when configuring WS-Security for WebCenter Services Portlets.

---

2. After creating the pagelet producer, you must then register it with Oracle WebCenter Interaction.

   For information about how to do this, see the "Creating the Oracle WebCenter Pagelet Producer Remote Server" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Interaction* at:

   `http://download.oracle.com/docs/cd/E23010_01/wci.1033/`

3. When the pagelet producer is registered with your Oracle WebCenter Interaction application, you must then create remote pagelet web services for each of the pagelets that you want to use.

   For information about how to do this, see the "Creating or Editing a Remote Pagelet Web Service" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Interaction* at:

   `http://download.oracle.com/docs/cd/E23010_01/wci.1033/`

4. Next, create portlets for the pagelets.

   For information about how to do this, see the "Creating or Editing a Portlet" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Interaction* at:

   `http://download.oracle.com/docs/cd/E23010_01/wci.1033/`

5. Finally, you can add the portlets to your Oracle WebCenter Interaction community pages.

   For information about how to do this, see the "Creating a Community Page" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Interaction* at:

   `http://download.oracle.com/docs/cd/E23010_01/wci.1033/`

64.7.4 Setting WebCenter Services Portlets Parameters

Each portlet provided by WebCenter Services Portlets defines various parameters that enable you to change the appearance or behavior of the portlet when you consume it in your application.

This section includes the following subsections:

- Section 64.7.4.1, "Common WebCenter Services Portlets Parameters"
- Section 64.7.4.2, "Document Manager Portlet Parameters"
- Section 64.7.4.3, "Discussion Forums Portlet Parameters"
- Section 64.7.4.4, "Blogs Portlet Parameters"
- Section 64.7.4.5, "Lists Portlet Parameters"
- Section 64.7.4.6, "Polls Manager Portlet Parameters"
- Section 64.7.4.7, "Take Polls Portlet Parameters"

- Section 64.7.4.8, "Worklist Portlet Parameters"
- Section 64.7.4.9, "Announcements Portlet Parameters"
- Section 64.7.4.10, "Mail Portlet Parameters"
- Section 64.7.4.11, "Activity Stream Portlet Parameters"
- Section 64.7.4.12, "Tag Cloud Portlet Parameters"

64.7.4.1 Common WebCenter Services Portlets Parameters

Table 64–9 lists the parameters that are available for all the portlets provided by WebCenter Services Portlets.

Table 64–9 Common Parameters for WebCenter Services Portlets

Parameter	Description
`width`	Specifies the width of the portlet on the consuming page. If not specified, the portlet takes up as much horizontal space as the page allows.
`height`	Specifies the height of the portlet on the consuming page. If not specified, the portlet height is determined by the portlet consumer.

64.7.4.2 Document Manager Portlet Parameters

Table 64–10 lists the additional parameters available for the Document Manager portlet.

Table 64–10 Document Manager Portlet Parameters

Parameter	Description
`connectionName`	The name of the Oracle Content Server connection used by WebCenter Services Portlets. If no value is selected, the default connection specified by the application developer or administrator is used. For information about configuring content repository connections, see the "Registering Content Repositories" section in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*. Default: The connection selected as default in the Create Content Repository Connection dialog box by the application developer, which can be changed by the administrator.
`featuresOff`	A list of disabled features for the portlet. Use commas or spaces to separate items. Valid values are exposed in the JavaDoc: `checkin, checkout, clipboard, close, delete, download, dnd, editwiki, editoffice, newfolder, newwiki, rename, upload, multifile-upload, profile-upload, search, advancedSearch, workflow, properties, history, comments, likes, links, tags, recommendations, autovue, title, related-items, social, sidebars, ils.`  Example:  `${'search, advancedSearch, clipboard, dnd, rename, newfolder, upload, newwiki, checkin, checkout, editoffice, edithtml, delete, sidebars, history'}`

Table 64–10 (Cont.) Document Manager Portlet Parameters

Parameter	Description
`layout`	A target layout for the task flow. Select from: ■ `${'explorer'}`—(default) Displays folders and files in two panes; the left pane shows folders, and the right pane show the contents of the currently selected folder ■ `${'table'}`—Displays only the contents of the current folder in a single pane, with the capability to click a folder to drill down, refreshing the pane with the folder contents ■ `${'treeTable'}`—Displays the folder hierarchy in a single pane, beginning with the root folder, with the capability to expand and collapse folders
`pageSize`	The maximum number of rows to show in the portlet. If the listing of folders and files in the portlet is larger than the specified number of rows, the portlet displays a scroll bar. Default: `27` **Note:** If you set `pageSize` to a value that is too big for the size of the screen, the end user will experience difficulty coordinating the portlet scroll bar with the application scroll bar.
`readOnly`	Specifies whether to disable and hide all content management operations: ■ `${true}`—Disable content management ■ `${false}`—(Default) Expose content management to users
`resourceId`	The currently focused resource. This value can be a folder ID or a document ID.
`startFolderPath`	The name of the folder to use as the root folder in the current portlet instance. This is a content-scoping parameter that assists with determining the source and range of content to display in the portlet instance. You can specify an EL expression to set this value. Example: `${'/WebCenterB5/Proj_X/Specs'}`

64.7.4.3 Discussion Forums Portlet Parameters

Table 64–11 lists the additional parameters available for the Discussion Forums portlet.

Table 64–11 Discussion Forums Portlet Parameters

Parameter	Description
`categoryId`	An identifier for an existing category in WebCenter Portal's Discussion Server to which the view should be scoped. When no value is supplied, it defaults to the appropriate root category of the discussions server. (This root category ID can be overridden by supplying an additional property named application.root.category.id in the connection.) For testing purposes, you may want to create a category through the discussions server administrator interface and then reference that category identifier here.
`forumId`	An identifier for an existing forum in your discussions server for which popular topics should be fetched. If both categoryId and forumId are given, then only categoryId is honored.

Table 64–11 (Cont.) Discussion Forums Portlet Parameters

Parameter	Description
`showRecursiveForums`	Determines if the portlet shows forums either in a category only or in subcategories. True means all forums under a given category/subcategory are shown; false means only the category's direct child forums are shown. The default value is false. Note: A value of true can impact performance.
`isCategoryView`	Determines if the forums are grouped under the Category ID category or the topics are grouped under the Forum ID forum. True means the portlet displays the forums classified under categoryId; false means the portlet displays the topics associated with the specified forumId. The default value is false. This parameter works in combination with other parameters.

64.7.4.4 Blogs Portlet Parameters

Table 64–12 lists the additional parameters available for the Blogs portlet.

Table 64–12 Blogs Portlet Parameters

Parameter	Description
`hideComments`	Specifies whether or not to display blog post comments: ■ true—Comments are shown for blog posts ■ false—(Default) Comments are hidden for blog posts
`pageSize`	Specifies how many blog posts to show before pagination is displayed and users can select to go to the next page. By default, up to 10 blog posts can be displayed on a blog page.
`resourceId`	Specifies the unique identifier of the selected folder. The value can be specified in the following formats: ■ *`connection_name/path_to_folder`* Where, *`connection_name`* is the name of the Oracle Content Server connection used by WebCenter Services Portlets, and *`path_to_folder`* is the path to the folder on Oracle Content Server that you want to expose as a blog. ■ *`connection_name`*`#dCollectionID:`*`dCollectionId`* Where, *`connection_name`* is the name of the Oracle Content Server connection used b WebCenter Services Portlets, and *`dCollectionId`* is the collection ID of the folder on Oracle Content Server that you want to expose as a blog.

64.7.4.5 Lists Portlet Parameters

The Lists portlet does not include any additional parameters.

64.7.4.6 Polls Manager Portlet Parameters

Table 64–13 lists the additional parameters available for the Polls Manager portlet.

Table 64–13 Polls Manager Portlet Parameters

Parameter	Description
`showUserDataOnly`	Determines whether to display all polls or only those polls created by the user. The default (`No`) is to show all polls. Set to `Yes` to display only those polls created by the user.

64.7.4.7 Take Polls Portlet Parameters

Table 64–14 lists the additional parameters available for the Take Polls portlet.

Table 64–14 Take Polls Portlet Parameters

Parameter	Description
`pollId`	The ID of the poll to display.

64.7.4.8 Worklist Portlet Parameters

The Worklist portlet does not include any additional parameters.

64.7.4.9 Announcements Portlet Parameters

Table 64–15 lists the additional parameters available for the Announcements portlet.

Table 64–15 Announcements Portlet Parameters

Parameter	Description
`parentId`	The forum ID in the discussions server under which announcement objects are maintained. Each application should create a forum on the discussions server. Enter that forum ID here, for example `${2}`. If this parameter is not specified, then amusements default to global announcements.

64.7.4.10 Mail Portlet Parameters

Table 64–16 lists the additional parameters available for the Mail portlet.

Table 64–16 Mail Portlet Parameters

Parameter	Description
`tabularView`	Using the EL value type, enter a value of true to display the information associated with a mail message, such as its subject, sender, and, date sent, in a tabular format. If this parameter is set to false, then mail messages render in a list view.

64.7.4.11 Activity Stream Portlet Parameters

Table 64–17 lists the additional parameters available for the Activity Stream portlet.

Table 64–17 Activity Stream Portlet Parameters

Parameter	Description
`advancedQuery`	A field for specifying a custom query to filter streamed items. For more information, see Section 40.3, "What You Should Know About the Activity Stream Advanced Query Option."

Table 64–17 (Cont.) Activity Stream Portlet Parameters

Parameter	Description
`enableContextInfo`	A means of showing or omitting detailed information about the object in the current context (that is, in a popup or other contextual instrument). ■ Enter `#{true}` to enable Activity Stream to display the `af:contextInfo` component (a small, red dot). Users can click this dot to view object detail information. ■ Enter `#{false}` to omit object detail information. When this parameter is true and the `contextInfoTaskflowId` in the service definition is defined, the activity from the service displays the `af:contextInfo`. Otherwise, no context information is shown.
`hideComments`	A means of showing or hiding the Comments feature on streamed activities. Enter `true` to hide the Comments feature. Enter `false` to show it.
`hideConfigure`	A means of hiding the personalization option on the task flow instance.
`hideInlinePreview`	A means of allowing or omitting an inline preview of files attached to streamed activities. Enter `true` to omit a file preview. Enter `false` to show it.
`hideLike`	A means of showing or hiding the Like link on a streamed activity. Enter `true` to hide the Like link. Enter `false` to show it.
`hideShare`	A means of showing or hiding the Share menu on streamed activities. Enter `true` to hide the Share menu. Enter `false` to show it.
`pageSize`	The number of items to stream in a given task flow instance.
`pagination`	The form of pagination to use on a multipage stream. ■ Enter `#{true}` to show Previous and Next links. ■ Enter `#{false}` hide to omit Previous and Next links. Instead, a more link is rendered, enabling users to navigate to a fuller view of the task flow where all streamed activities are shown.
`profileOnly`	A means of streaming activities only from user profiles. ■ Enter `true` to stream only those activities associated with user profiles ■ Enter `false` to stream other types of activities along with those associated with user profiles
`resourceId`	The current user ID. Enter `#{securityContext.userName}` to return the current user.

Table 64–17 (Cont.) Activity Stream Portlet Parameters

Parameter	Description
`serviceCategories`	A field for entering a comma-separated list of names of services from which to stream activities. Use this parameter to limit the display of streamed activities to only those associated with the specified service or services. Enter one or more service IDs, for example: `oracle.webcenter.collab.announcement, oracle.webcenter.collab.forum` For a list of valid service ID, see Table G–7. Note that all listed service IDs cannot be used because all services do not stream items to the Activity Stream. For example, the RSS service does not stream any activities.

64.7.4.12 Tag Cloud Portlet Parameters

The Tag Cloud portlet does not include any additional parameters.

64.8 Adding the Portlet Producer Task Flow

The portlet producer task flow (Producer) enables users to manage portlet producer connections at runtime.

For applications created using WebCenter Portal's Framework application template, the portlet producer task flow is available out of the box through the WebCenter Portal Administration console (Services tab). For details, see the chapter "Managing Services, Portlet Producers, and External Applications" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

In addition, just like other task flows, you can add the portlet producer task flow to your application pages. This might be especially useful if you are not using WebCenter Portal's Framework application template and the WebCenter Portal Administration console is therefore not part of your project. Special permissions are required to manage or view portlet producer connections through the Producer task flow:

- **`AppConnectionManager`**—Users with this role can register, modify, and delete portlet producer connections at runtime.
- **`AppConnectionViewer`**—Users with this role can view portlet producer connections at runtime. By default, any user who is logged in (that is, has the `authenticated-user` role) is granted this role.

By default, users with the Administrator role can manage portlet producers. If you want other users to manage connections through the portlet producer task flow, you must grant them the `AppConnectionManager` role.

To add the portlet producer task flow:

1. Create or open a JSF page in your application where you want the task flow to be added (see Section 5.3, "Adding Pages to a Portal").
2. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
3. Drag **Producer** from the Resource Palette and drop it onto the page inside the `af:form` tag.
4. Grant the `AppConnectionManager` role to one or more test users, if required:
   a. Add the test user `TEST_PROD`.

b. Grant the `AppConnectionManager` role.

For information about how to add a user and grant this role, see the section "Creating Test Users" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

5. Save and run your page. Login as an administrator or the `TEST_PROD` user defined in the previous step. The screen shown in Figure 64–10 appears.

Figure 64–10 The Portlet Producer Task Flow



Note: At runtime, application administrators can grant users the `AppConnectionManager` and `AppConnectionViewer` roles through WebCenter Portal Administration (see the section "Adding Members to Application Roles" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*). Alternatively, system administrators can grant `AppConnectionManager` and `AppConnectionViewer` roles through Fusion Middleware Control (see "Granting Application Roles Using Fusion Middleware Control" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*).

64.9 Troubleshooting Portlets

This section provides information to assist you in troubleshooting problems you may encounter while using portlets. It contains the following subsections:

- Section 64.9.1, "Diagnostic Tools for Troubleshooting Portlets"
- Section 64.9.2, "Configuring the Portlet Logging File"
- Section 64.9.3, "Portlet Displays a Portlet Consumer Error"
- Section 64.9.4, "Portlet Displays a Portlet Timeout"
- Section 64.9.5, "Portlet Displays a Remote Portlet Communication Error"
- Section 64.9.6, "Portlet Displays a Remote Portlet Error"

64.9.1 Diagnostic Tools for Troubleshooting Portlets

There is a set of tools available for both the consumer and producer to help identify and resolve issues with portlets.

If you encounter a portlet error message when a portlet is rendered, or if the portlet displays but you cannot interact correctly with it, there are some general steps using these tools that you should follow to diagnose the issue.

This section includes the following subsections:

- Section 64.9.1.1, "Identify the Portlet Instance"

- Section 64.9.1.2, "Examine the Portlet Consumer Test Page"
- Section 64.9.1.3, "Examine the Producer Test Page"

64.9.1.1 Identify the Portlet Instance

The first step when you encounter a portlet error, is to identify which portlet producer and portlet instance is being invoked. Execute the `portletDebugShow()` JavaScript from your browser to display this information in the main portlet content area.

To identify the portlet instance:

1. Enter the following command in the Location field of your browser:

   ```
   javascript:portletDebugShow()
   ```

   Tip: In Internet Explorer and Google Chrome, you must type this command in the Location field. If you paste the command into the field, the `javascript` piece is removed.

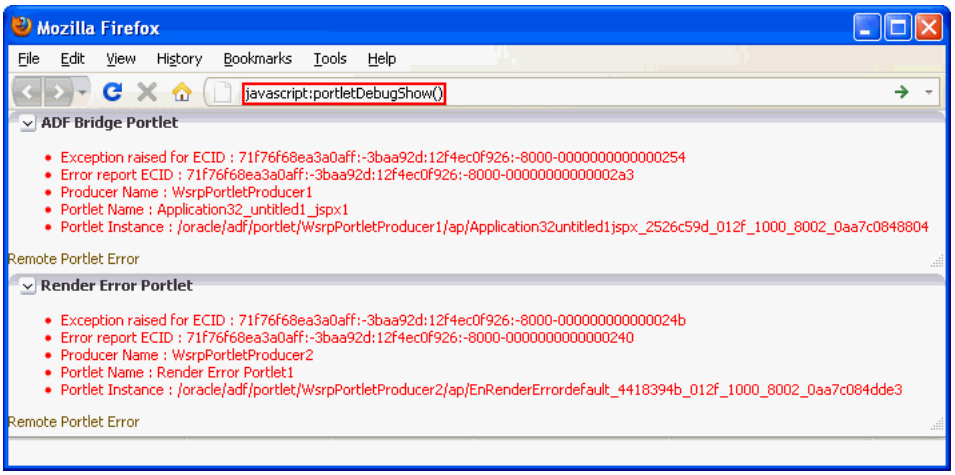
   In Firefox 6 and above, you cannot enter JavaScript in the Location field, you must enter the command in JavaScript console.

2. After running the script, every portlet now displays the following information:
   - Producer name
   - Portlet name
   - Portlet instance ID
   - Execution Context IDs (ECIDs)

     The ECIDs are unique IDs used to identify a portlet request. Use the ECIDs to correlate the messages across different consumer and producer log files using Fusion Middleware Control. The same ECID is propagated from the consumer to the producer. For more information, see the "Correlating Messages Across Log Files and Components" section in the *Oracle Application Server Administrator's Guide*.

     Note: Broken portlets show two ECIDs: one for the request in which the error occurred and one for request in which the error was reported. For inline portlets (that is, portlets that are not displayed in an IFRAME), these two ECIDs are the same.

     For IFRAME portlets, for example Oracle JSF Portlet Bridge portlets, the ECIDs are different. This is because the error is reported in a later request than the one in which the original exception occurred. When checking the logs, you should look for both ECIDs, as either may contain relevant information.



You can use this information in the subsequent diagnostic steps to help locate the issue.

Note: The ECIDs shown in the portlet diagnostic information do not reflect partial page rendering requests that have been made to the portlet producer (using the portlet consumer resource proxy). These requests may update the portlet, but the ECIDs are not recorded in the portlet diagnostic information. Errors that occur during these requests are logged on the producer and by the portlet resource proxy on the consumer but you cannot use the ECID information reported in the portlet diagnostic information to help you determine the ECIDs for the relevant log entries.

---

3. When you have finished debugging the portlets, enter the following command to hide the portlet debugging information:

```
javascript:portletDebugHide()
```

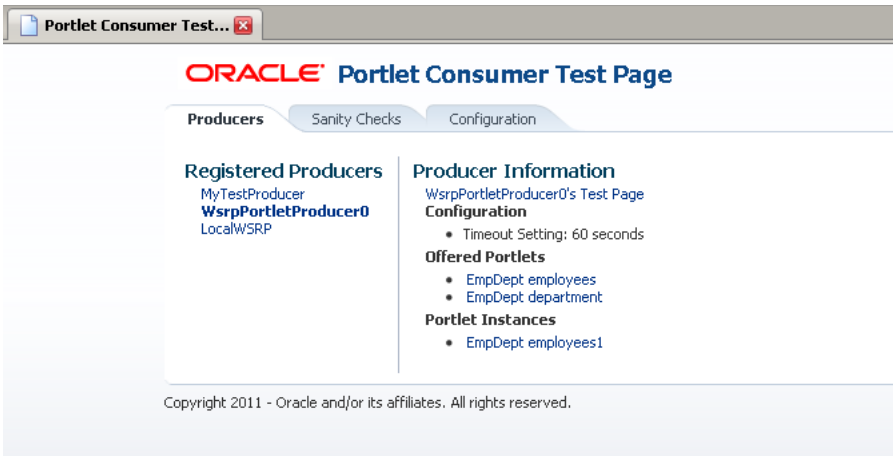
Tip: In Internet Explorer and Google Chrome, you must type this command in the Location field. If you paste the command into the field, the `javascript` piece is removed.

In Firefox 6 and above, you cannot enter JavaScript in the Location field, you must enter the command in JavaScript console.

64.9.1.2 Examine the Portlet Consumer Test Page

The next step in diagnosing a portlet error is to access the Portlet Consumer Test Page (shown in Figure 64–11) to locate the portlet producer and, if necessary, test the portlet in isolation.

Figure 64–11 The Portlet Consumer Test Page



The Portlet Consumer Test Page contains three tabs:

- **Producers:** This tab lists all the producers registered with the consumer application. Selecting a producer provides specific information about that producer.
- **Sanity Checks:** This tab may contain a predefined set of portlet instances and required parameters that can be run in the consumer application, as configured by the consumer application developer. Any failures within these portlets indicate a problem with the corresponding producer and/or portlet.
- **Configuration:** This tab enables you to identify the consumer configuration entries for portlet consumption. You cannot change these values as they are stored within the application; they are displayed for reference information only.

After accessing the Portlet Consumer Test Page, you can perform further diagnostic steps.

This section includes the following subsections for using the Portlet Consumer Test Page to diagnose portlet issues:

- Section 64.9.1.2.1, "Access the Portlet Consumer Test Page"
- Section 64.9.1.2.2, "Locate the Portlet Producer"
- Section 64.9.1.2.3, "Locate and Run the Portlet Instance"
- Section 64.9.1.2.4, "Perform Sanity Checks"
- Section 64.9.1.2.5, "Check Consumer Configuration Entries"

64.9.1.2.1 Access the Portlet Consumer Test Page

The Portlet Consumer Test Page provides diagnostic information about the portlet consumer.

To access the Portlet Consumer Test Page:

1. In your browser, enter the URL for the Portlet Consumer Test Page:

   `http://host:port/context-root/faces/oracle/portlet/client/adf/diagnostic/pages/ConsumerTestPage.jspx`

Note: If the consumer application is secured, the Portlet Consumer Test Page can be accessed only by users granted permission to view those pages.

2. In the Portlet Consumer Test Page, you can perform further diagnostic steps as described in the following sections.

64.9.1.2.2 Locate the Portlet Producer

The Producers tab of the Portlet Consumer Test Page lists all the producers that have been registered with the consumer application. If a portlet instance in your application displays an error message, you can view information about the producer that owns the portlet by selecting it on this tab.

To locate the portlet producer:

1. In the Portlet Consumer Test Page, select the portlet producer that owns the portlet instance that is reporting the error.

   You noted this information in Section 64.9.1.1, "Identify the Portlet Instance."

2. The following information is provided for the selected producer:

   - Producer Test Page: A link to the Producer Test Page.
   - Configuration: Details of potential issues surrounding skins, security, and timeouts associated with the using producer.
   - Offered Portlets: A list of all portlets offered by the producer. If there are no offered portlets listed, this indicates that there is a problem with the registration metadata for the producer.
   - Portlet Instances: A list of all portlet instances for the selected producer in the consumer application. This list may be empty.

   You can use this information to identify potential issues with the producer.

64.9.1.2.3 Locate and Run the Portlet Instance

If you have still not been able to identify the cause of the portlet error, the issue may lie with the portlet instance itself.

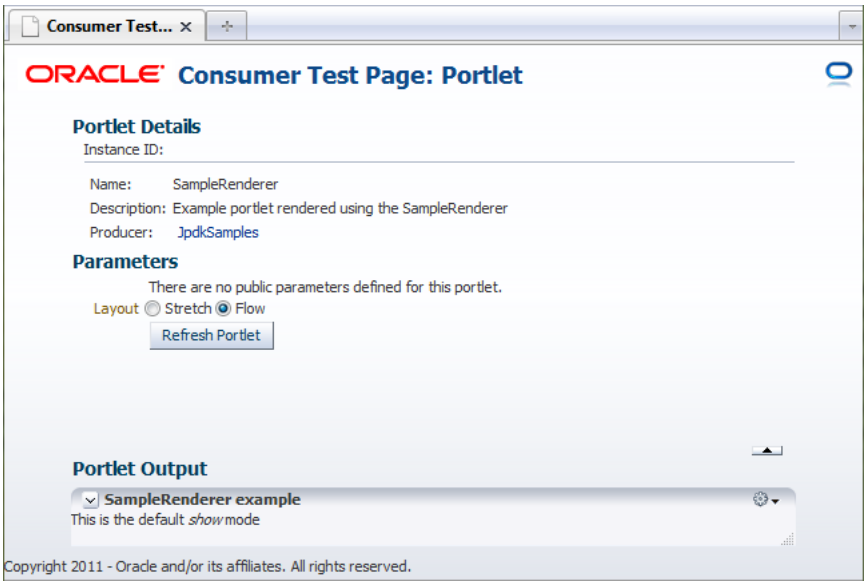
To locate and run the portlet instance:

1. In the Portlet Consumer Test Page, select the portlet producer that owns the portlet instance that is reporting the error.

   You noted this information in Section 64.9.1.1, "Identify the Portlet Instance."

2. Under Portlet Instances, select the portlet instance to display the Consumer Test Page: Portlet page.

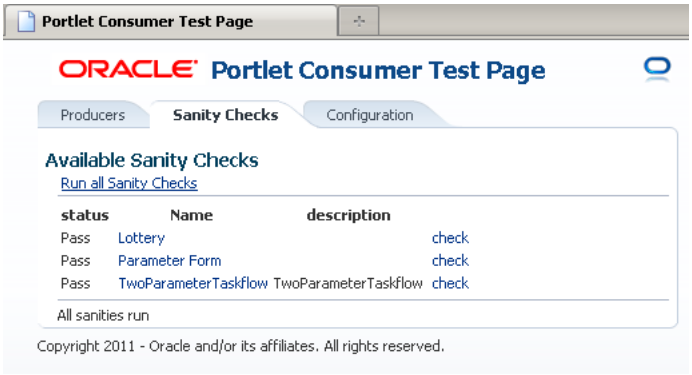
   You noted this information in Section 64.9.1.1, "Identify the Portlet Instance."



3. The Portlet Consumer Test Page: Portlet page renders the portlet in a standalone page. If the portlet runs correctly on this page, the problem is most likely caused by other components on the page containing the broken portlet.
4. The Parameters section enables you to experiment with how the portlet looks using a stretch or flow layout
5. If the portlet contains parameters, the Parameters section also lists all the public parameters for the portlet. Enter values for any parameters to test that the portlet is receiving parameters correctly.
6. To navigate back to the Portlet Consumer Test Page, click the producer name link at the top of the page.

64.9.1.2.4 Perform Sanity Checks The Sanity Checks tab of the Portlet Consumer Test Page (shown in Figure 64–12) provides a quick overview of the state of portlet communication in your application across all producers.

Figure 64–12 The Sanity Checks Tab



The Sanity Checks tab references portlet instances used within the consumer application. This list is configured by the application developer who chose the portlets to include and the parameters to pass to these portlets.

The checks on this page do not render the output in the UI, but simply create a runnable instance of the portlet under the covers and report any failures if any exception is returned by the portlet.

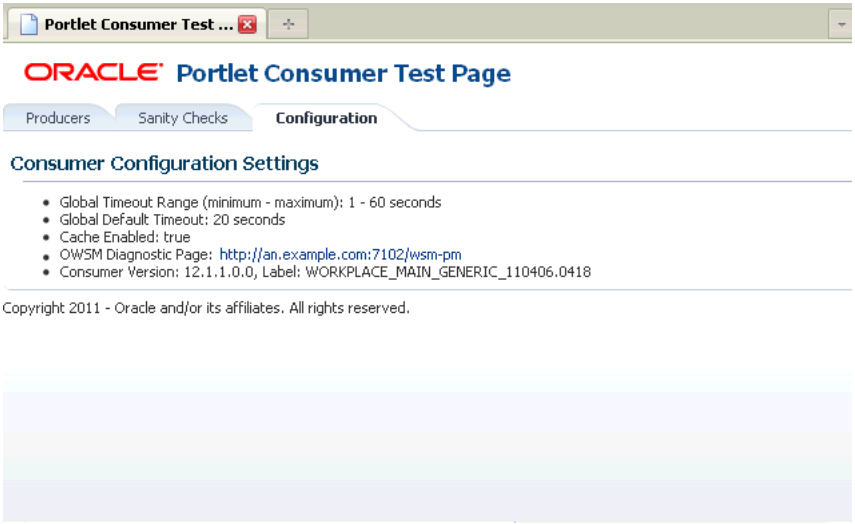
To perform sanity checks:

1. In the Portlet Consumer Test Page, click the **Sanity Checks** tab.
2. Click the **check** link next to the portlet that you want to test.

   The results of the sanity tests are displayed in the **Status** column.
3. To run sanity checks on all listed portlets, click the **Run all Sanity Checks** link.

64.9.1.2.5 Check Consumer Configuration Entries The Configuration tab of the Portlet Consumer Test Page (shown in Figure 64–13) enables you to identify the consumer configuration entries for portlet consumption. This tab displays settings defined in the `adf-config.xml` file, for example, the minimum and maximum timeout values and the consumer version number. You cannot change these values as they are stored within the application; they are displayed for reference information only.

Figure 64–13 The Configuration Tab



64.9.1.3 Examine the Producer Test Page

If you cannot identify the cause of the error in the consumer application, the next step is to use the Producer Test Page (shown in Figure 64–14) to identify potential issues with the portlet producer application.

Figure 64–14 The Producer Test Page



Access to the main Producer Test Page is public, but links to the test pages for each portlet are accessible only to users granted permission on the underlying pages and task flows.

The Producer Test Page contains five sections:

- Portlets

  A list of all the portlets within the producer. For Oracle JSF Portlet Bridge portlets, each portlet also provides a separate link to run the portlet as a servlet (this is a prerequisite to running them as portlets: if a portlet does not run as a servlet, it cannot run as a portlet).

- Container Configuration

  Information on where the consumer preference information is stored.

- Container Version

  The version number of the Portlet Producer Container.

- WSDL URLs

  Links to the Web Service Definition Language (WSDL) documents to use for registration.

- SOAP Monitor

  A link to the WSRP SOAP monitor where users with the `Monitor` or `Admin` role can track the SOAP messages between the consumer and producer.

After accessing the Producer Test Page, you can perform further diagnostic steps.

This section includes the following subsections:

- Section 64.9.1.3.1, "Access the Producer Test Page"

- Section 64.9.1.3.2, "Run the JSF Portlet as a Servlet"
- Section 64.9.1.3.3, "Examine the SOAP Monitor"

64.9.1.3.1 Access the Producer Test Page The Producer Test Page provides diagnostic information about the portlet producer.

To access the Producer Test Page:

1. In your browser, enter the URL for the Producer Test Page:

   `http://host:port/context-root/info`

2. In the Producer Test Page, you can perform further diagnostic steps as described in the following sections.

64.9.1.3.2 Run the JSF Portlet as a Servlet To verify that an Oracle JSF Portlet Bridge portlet producer is running correctly, you must first verify that the producer application runs correctly through standard HTTP requests. If the artifacts the producer exposes as portlets do not run as servlets, they will not run as portlets.

To run a JSF portlet as a servlet:

1. In the Producer Test Page, click the **run as servlet** link next to the portlet.
2. The portlet is called using standard HTTP to request the underlying page or task flow. The results of the request are displayed in a new browser window.

   If the resulting page or task flow does not render correctly, then there is a problem with the producer application that must be resolved before you can run the page or task flow as a portlet.
3. If the portlet accepts parameters, click **show parameters** to list them and provide values. When you click **run as servlet**, the portlet call includes the parameter values.

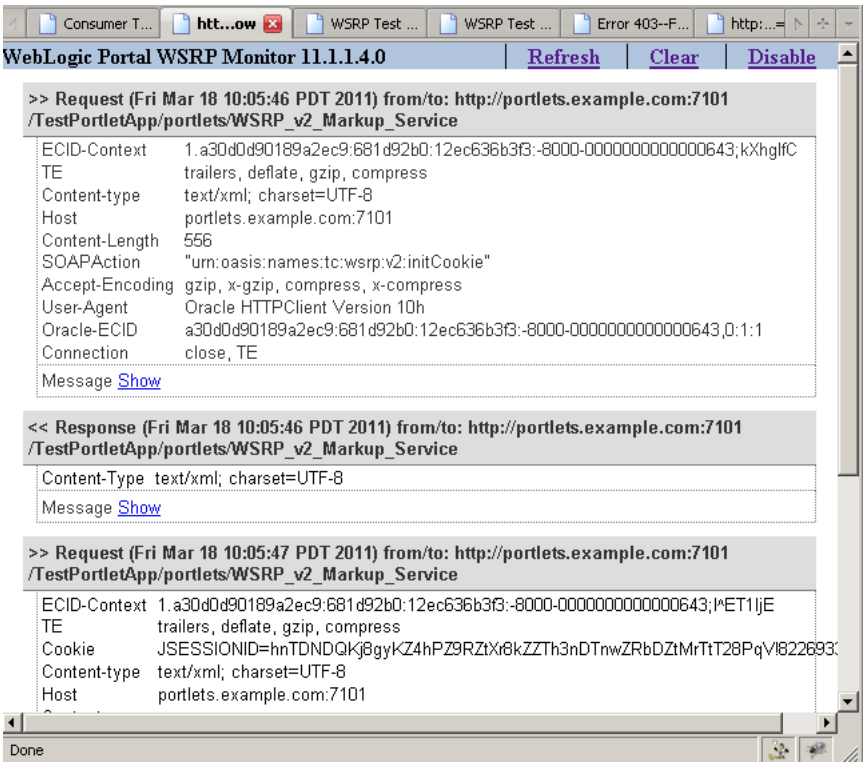
64.9.1.3.3 Examine the SOAP Monitor The SOAP monitor provides access to the SOAP requests between the consumer and producer when rendering a portlet. This is very useful in diagnosing problems at the communication level.

To examine the SOAP monitor:

1. In the Producer Test Page, click the SOAP Monitor link at the bottom of the page.
2. When prompted, enter your user name and password.

   Note: To access the SOAP monitor you must be a member of the `Monitors` or `Administrators` role in the Identity Management System.

3. By default, the SOAP monitor is disabled, so the page is empty. You must first enable the monitor by clicking the **Enable** link at the top of the page.
4. The page does not automatically refresh, so to display SOAP messages, you must click the **Refresh** link.
5. To force a request to the failing portlet, go to the Portlet Consumer Test Page: Portlet page for the portlet and select **Refresh Portlet**.
6. When the portlet has rendered, or failed, click the Refresh link in the SOAP monitor to display the captured request.



7. Now, you can investigate the SOAP messages that were sent and the responses to try to narrow down the cause of the problem.

> **Note:** If, after rerunning the portlet and refreshing the SOAP monitor, you see no messages displayed, this indicates that there may be a security issue between the producer and the consumer. You must verify that the correct WS-Security settings are set up for the producer and consumer to communicate.

64.9.2 Configuring the Portlet Logging File

To troubleshoot portlet issues, it is useful to add portlet log-handlers and loggers to the logging configuration file, `logging.xml`.

Example 64–14 shows how to add the portlet log-handlers and loggers. The example assumes that you are running the consumer and producer applications on the same WebLogic Server instance. If you are running the consumer and producer applications on different instances, you must split them up appropriately.

> **Note:** Add the log entries at the end of the file to ensure that they override any seeded settings.

Example 64–14 Configuring Log Files for Troubleshooting Portlet Issues

```
<!-- NOTE: You need to change the path where the logfile is located -->
<log_handlers>
...
   <!-- Portlet Consumer -->
   <log_handler name="portlet-consumer-handler" class="oracle.core.ojdl.logging.ODLHandlerFactory">
      <property name="format" value="ODL-Text"/>
```

```
      <property name="path" value="/scratch/logs/portlet-consumer.log"/>
   </log_handler>

   <!-- Portlet Producer -->
   <log_handler name="portlet-producer-handler" class="oracle.core.ojdl.logging.ODLHandlerFactory">
      <property name="format" value="ODL-Text"/>
      <property name="path" value="/scratch/logs/portlet-producer.log"/>
   </log_handler>

   <!-- Portlet Bridge -->
   <log_handler name="portlet-bridge-handler" class="oracle.core.ojdl.logging.ODLHandlerFactory">
      <property name="format" value="ODL-Text"/>
      <property name="path" value="/scratch/logs/portlet-bridge.log"/>
   </log_handler>
...
</log_handlers>

<loggers>
...
   <!-- Portlet Consumer -->
   <logger name="oracle.portlet.client" level="FINEST" useParentHandlers="false">
      <handler name="portlet-consumer-handler"/>
   </logger>

   <!-- Portlet Servers -->
   <logger name="com.bea.portlets" level="FINEST" useParentHandlers="false">
      <handler name="portlet-producer-handler"/>
   </logger>
   <logger name="com.bea.netuix" level="FINEST" useParentHandlers="false">
      <handler name="portlet-producer-handler"/>
   </logger>
   <logger name="com.bea.wsrp" level="FINEST" useParentHandlers="false">
      <handler name="portlet-producer-handler"/>
   </logger>
   <logger name="oracle.portlet.producer" level="FINEST" useParentHandlers="false">
      <handler name="portlet-producer-handler"/>
   </logger>

   <!-- Portlet Bridge -->
   <logger name="oracle.portlet.bridge" level="FINEST" useParentHandlers="false">
      <handler name="portlet-bridge-handler"/>
   </logger>
   <logger name="oracle.portlet.server.bridge" level="FINEST" useParentHandlers="false">
      <handler name="portlet-bridge-handler"/>
   </logger>
...
</loggers>
```

The logging configuration file is located in:

DOMAIN_HOME/config/fmwconfig/servers/*server*/logging.xml

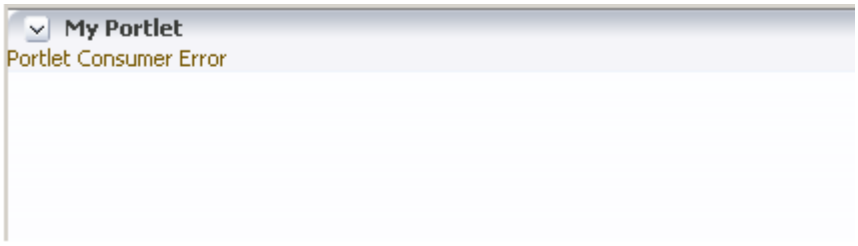
The log file name is also defined in `logging.xml`. By default the log file name is:

DOMAIN_HOME/servers/*server*/logs/*server*-diagnostic.log

64.9.3 Portlet Displays a Portlet Consumer Error

The message **Portlet Consumer Error** (shown in Figure 64–15) typically indicates that an error occurred within the operation of the portlet parts of the portlet consumer application.

Figure 64–15 Portlet Displaying a Portlet Consumer Error



Problem

An error has occurred within the operation of the portlet parts of the portlet consumer application. In other words, the error is unrelated to the remote portlet producer application.

Solution

Consult the diagnostic log file to determine the cause of the exception.

If the `DebugErrorRenderer` is enabled, the cause exception is displayed in the portlet along with links to the log file. If running in production mode, then consult the consumer-side logs.

The exception that caused the error message to be displayed is logged. Wherever possible, a message is included in the log at the start of the exception stack to indicate for which portlet binding the exception occurred. Example 64–15 shows a message logged for a portlet error.

Example 64–15 Example Message Logged for a Portlet Error

```
<PortletRenderer> <setErrorState> An error has occured for Portlet Binding
portlet1.
oracle.portlet.client.container.PortletContentTypeException: Unexpected content
type "null" in WSRPGetMarkup response.
...
```

Pay particular attention to the cause exceptions in the stack as this is likely to indicate what the real underlying problem is.

The cause is likely to be an internal error and the appropriate course of action is to contact Oracle Support.

64.9.4 Portlet Displays a Portlet Timeout

If a **Portlet Timeout** is displayed in the area of the page that you would expect to contain a portlet (as shown in Figure 64–16), this means that the consumer waited for a configured period of time for the producer to respond and did not get a response during that time, or the response did not complete during that time. There are a number of possible causes.

Figure 64–16 Portlet Displaying a Portlet Timeout Error



Problem 1

The producer machine is overloaded.

Solution 1

Check the load on the producer managed server (the tools used to do this vary depending on the operating system that is running on the producer). If the load is high, check whether a particular process is causing this high load, and whether such a process could be run on another machine, or at a less busy time. If no single process is causing the high load, or if the Oracle WebLogic Server is causing the high load, and if the load is consistently high, consider whether the producer hardware is adequate, or whether it is necessary to upgrade it (or add further nodes to the cluster). Also consider adjusting the Oracle WebLogic Server configuration to increase the size of the request thread pool. For more information, see the Oracle WebLogic Server documentation.

Problem 2

The network is overloaded, or there are problems with the network affecting communication between the consumer and producer.

Solution 2

Check that you can ping the producer machine from the consumer machine. Check that you can access the producer's WSRP Producer Test Page in your local browser. If this works, check that you can access this same page from a browser running on the consumer machine. If any of these steps cause problems, and the machine is not overloaded, this could be a network problem, which should be investigated by a system administrator.

Problem 3

There is a deadlock (or a stuck thread) on the producer machine causing the request thread to hang.

Solution 3

This should not happen during normal operation. If it does occur, there will generally be an error in the producer's log files indicating the point at which the deadlock occurred. This may help diagnose the problem. In some cases, it may be possible to alleviate this by modifying the configuration of Oracle WebLogic Server. For more information, see the Oracle WebLogic Server documentation.

Problem 4

The producer application is running slowly (for example, due to processing large quantities of data).

Solution 4

In this case, the producer application may be processing large quantities of data, causing it to spend too long building the response. If the application will regularly deal with large quantities of information, it may be necessary to either add or improve producer hardware, or to increase the portlet timeout duration. The portlet timeout can be configured on the producer connection in the consumer application using Enterprise Manager or the WLST `setWSRPProducerConnection` command. Additionally, minimum and maximum timeouts for all producer connections within the application can be configured within the portlet section of the `adf-config.xml` file.

Problem 5

The producer application is waiting for a response from another resource, such as a database, that is taking too long (for example, if the database is overloaded).

Solution 5

Check that the resource in question is functioning correctly. If it is, the solution is same as Solution 4.

Problem 6

The portlet timeout values have been misconfigured such that the timeout period is too short.

Solution 6

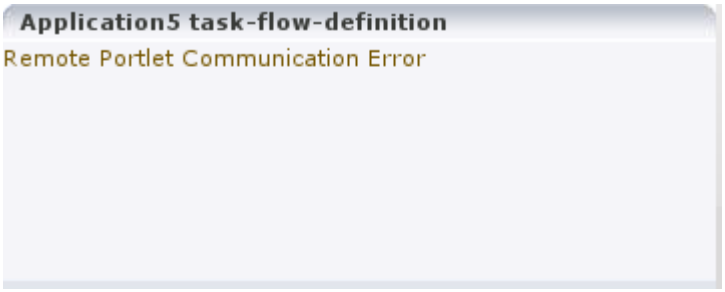
Typically, the timeout for a portlet is set on the registration of the producer. This may have been set to a value that does not give time for the portlet to respond.

Also, the portlet section of the `adf-config.xml` file allows minimum, maximum, and default values for portlet timeouts to be configured across the whole application. The maximum timeout imposes an upper limit on timeouts specified by portlet producers, so if the maximum timeout is too short, this could cause unwanted portlet timeout errors even if the timeout specified on the producer connection is longer.

64.9.5 Portlet Displays a Remote Portlet Communication Error

When a section of the screen shows the **Remote Portlet Communication Error** message (as shown in Figure 64–17), and there is an otherwise blank region surrounding it, this area is expected to be filled with a portlet, which the application is not able to contact.

Figure 64–17 Portlet Displaying a Remote Portlet Communication Error



Problem 1

The producer is down.

Solution 1

It could be that the producer application is not running, or the managed server on which it is deployed is not started. In this case, it will need to be started. Identify the application that needs to be started based on the task being attempted at the time of the portlet failure.

Problem 2

The web services security is incorrectly configured.

Solution 2

Troubleshooting steps for web services security depend on the type of security profile being used, for example AuthN, SSL, or Message Security.

For more information about troubleshooting web service security, see the "Diagnosing Problems" chapter in the *Oracle Application Server Web Services Security Guide*.

Problem 3

The producer managed server cannot be reached.

Solution 3

The producer may be in a location that cannot be reached by the consumer application, due to intervening firewalls or incorrect routing rules. In an environment that is installed by Oracle's provisioning software, this should not be the case, but it is worth checking that you are able to access the WSDL endpoint for the producer from the machine hosting the consumer, by going to:

`http://host:port/context-root/portlets/wsrp2?WSDL`

Where:

- *`host`* is the server to which the portlet producer is deployed
- *`port`* is the port to which the server is listening for HTTP requests
- *`context-root`* is the producer web application's context root

For example:

```
http://portlets.example.com:9999/sample/portlets/wsrp2?WSDL
```

64.9.6 Portlet Displays a Remote Portlet Error

If the portlet displays a **Remote Portlet Error** (as shown in Figure 64–18), this indicates that the producer responded with an error message. The error message is returned in the form of a SOAP fault message inside the response document. There are a number of reasons the producer might return an error. The best strategy to diagnose these errors is to first find the corresponding exception stack trace in the consumer diagnostic logs. This stack trace shows what kind of fault was returned by the producer, plus any further information required in the response. Some faults you may encounter are listed in the following sections.

Figure 64–18 Portlet Displaying a Remote Portlet Error



Problem 1

`OperationFailedException`. This is the most common type of Remote Portlet Error and it is a catch-all for most unhandled exceptions raised in the producer application.

Solution 1

To resolve an `OperationFailedException`, examine the exception in the consumer diagnostic logs. This generally shows any exception that was raised in the producer application to trigger the fault response as the final `Caused by` exception.

If required, you can then examine the diagnostic logs on the producer application for more detail, or for any related exceptions that occurred prior to the fault being triggered. In some cases, the exception in the producer log indicates a problem that can be simply resolved, such as a database connection failure or configuration problem. In other cases, the exception might indicate a product bug.

Problem 2

`InvalidRegistrationException`. This error indicates that the producer has not been properly registered with the consumer before the consumer attempted to communicate with it. This could also occur if the producer's preference store has been moved or deleted since the consumer registered it.

Solution 2

The most likely cause is a problem during provisioning. It is also worth checking the producer application's `web.xml` file setting to ensure that the entry shown in Example 64–16 is present.

Example 64–16 Persistent Store Setting in web.xml

```
<env-entry>
  <env-entry-name>oracle/portal/wsrp/server/persistentStore</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>Consumer</env-entry-value>
</env-entry>
```

Problem 3

`InvalidHandleException`. This indicates that the consumer has asked the producer to render, or otherwise interact with, a portlet instance that the producer does not know about. This could occur if the producer's preference store has been corrupted in some way since the portlet was added to the page.

Solution 3

This error is most likely caused by a problem during provisioning, or a missing `persistentStore` setting in the `web.xml` file, as described in Solution 2.

Problem 4

`AccessDeniedException`. This indicates that the producer application decided that the current user did not have access to the portlet or task flow in question.

Solution 4

This could either be a legitimate error message or an indication of a configuration problem. In most cases, WebCenter Portal should deal with authorization errors gracefully, without a Portlet Remote Error being displayed.

65

Creating Portlets with OmniPortlet

OmniPortlet is a data publishing portlet that you add to your application at design time, and customize at runtime. It provides a runtime, wizard-based experience to allow page designers to publish data from a variety of data sources, including SQL, XML, web services, spreadsheets, and web pages to several different layouts, such as customizable charts and tables.

This chapter covers the information you may find useful if you are planning to use OmniPortlet in your Oracle JDeveloper environment. For information on how to use the OmniPortlet wizard, refer to *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This chapter includes the following sections:

- Section 65.1, "Introduction to OmniPortlet"
- Section 65.2, "Adding OmniPortlet to Your Application"
- Section 65.3, "Customizing OmniPortlet"
- Section 65.4, "Troubleshooting OmniPortlet Problems"

For information about registering and configuring OmniPortlet, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application" and Section E.1, "OmniPortlet Configuration Tips."

Note: You can find more information about developing different types of portlets and information about producers and other portlet technologies in Chapter 58, "Overview of Portlets."

65.1 Introduction to OmniPortlet

OmniPortlet is a subcomponent of Oracle WebCenter Framework that enables developers to easily publish data from various data sources using a variety of layouts without writing any code. You can base an OmniPortlet on almost any kind of data source, such as a web service, a SQL database, spreadsheet (character-separated values), XML, and even application data from an existing web page. Figure 65–1 shows an example of a portlet created using OmniPortlet.

Figure 65–1 Example of an OmniPortlet



OmniPortlet enables the WebCenter Portal: Framework application developer and component developer to do the following:

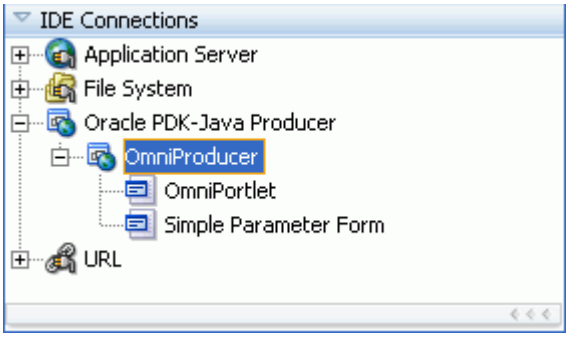
- Display data from multiple sources (CSV, XML, web service, SQL, and so on)
- Sort the data to display
- Format data using a variety of layouts (bulleted list, chart, HTML, and so on)
- Use portlet parameters
- Expose personalizable settings to page viewers

To display personalized data, you can refine the retrieved data by filtering the results returned from a data source, and parameterize the credential information used to access secure data. Out of the box, OmniPortlet provides the most common layout for portlets: tabular, chart, HTML, news, bulleted list, and form.

65.2 Adding OmniPortlet to Your Application

As described in Chapter 64, "Consuming Portlets," you can add an OmniPortlet to a page created through Oracle JDeveloper. OmniPortlet is included in the Integrated WebLogic Server (Integrated WLS) that is installed with Oracle JDeveloper. After you start the Integrated WLS, you can register the OmniPortlet producer by using the Register Oracle PDK-Java Producer wizard. When this producer is registered, the portlets become available on the Oracle JDeveloper Resource Palette or in the Application Resources panel. For example, if you register the OmniPortlet producer as "OmniProducer," OmniPortlet displays in your IDE Connections list in the Resource Palette, as shown in Figure 65–2.

Figure 65–2 OmniPortlet in the Resource Palette



Note: For more information about:

- Registering the producer, see Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application."
- Installing, initializing, and stopping the default server and pointing Oracle JDeveloper to it, see Section 3.4, "Working with the Integrated WebLogic Server."
- Adding an instance of OmniPortlet to your page, see Section 64.3, "Adding Portlets to a Page."

After you register the portlet producer, you can simply drag OmniPortlet onto your `*.jspx` page.

Note: When you add an instance of OmniPortlet onto your page in Oracle JDeveloper, open the Property Inspector for the portlet and ensure that the `AllModesSharedScreen` and `RenderPortletInIFrame` properties are set as follows:

- **AllModesSharedScreen** is set to `False` to display the Customize and Personalize in full page size.
- **RenderPortletInIFrame** is set to `True` to display the OmniPortlet in its own inline frame (IFRAME) in the View mode.

For information on configuring OmniPortlet in Oracle JDeveloper, refer to Section E.1, "OmniPortlet Configuration Tips."

65.3 Customizing OmniPortlet

After you add an OmniPortlet to your application at design time, you can customize the content, layout, and other options, by running your application to a browser. This section provides a high-level introduction to the runtime customization experience. For more detailed information on using and customizing this portlet, refer to *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

Note: For more information about configuring OmniPortlet, see Appendix E.1, "OmniPortlet Configuration Tips."

The OmniPortlet wizard initially contains five steps. When you first define your OmniPortlet, you set the data source type, data source options, filter options, view options, and layout. When you have completed these steps of the wizard, you can reenter the wizard by clicking the **Customize** link for the portlet. When you reenter the wizard, you can change the definitions on the Source, Filter, View, and Layout tabs.

Note: On the IBM Linux on Power platform, if the action buttons (Next, Previous, Finish, and Cancel) are minimized to dots when defining the OmniPortlet, increase the stack size shell limit to unlimited and restart the `WLS_portlet` instance. Run the following command to set the stack size shell limit to unlimited: `prompt> ulimit -s unlimited`.

You can use several different data sources with OmniPortlet:

- Spreadsheet
- SQL
- XML
- Web Service
- Web Page

For more information on using these data sources with OmniPortlet, refer to the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

65.4 Troubleshooting OmniPortlet Problems

This section provides information to help you troubleshoot problems you may encounter while using OmniPortlet.

Note: As the OmniPortlet producer exists and executes in a tier different from the Framework application and does not have access to the session information, you must expose XML files as PUBLIC in order for OmniPortlet to access them.

Cannot Define OmniPortlet Using the Define Link

You are not able to define the OmniPortlet at runtime.

Problem

OmniPortlet only supports a `RenderPortletInIFrame` value of `true`, meaning that OmniPortlet must be rendered within an IFRAME. Therefore, the OmniPortlet property, `RenderPortletInIFrame`, must be set to `true`. In the Property Inspector, the `RenderPortletInIFrame` property is available under Display Options.

Currently, the `RenderPortletInIFrame` property has a value of `false`. Consequently, when you click the **Define** link at runtime, the **Type** tab may not display and you cannot proceed with defining the OmniPortlet.

Solution

You can choose **Customize** from the Action list to define OmniPortlet, or select the OmniPortlet in the Structure window in Oracle JDeveloper, and in the Property Inspector, set `RenderPortletInIFrame` to `true`.

66

Creating Content-Based Portlets with Web Clipping

Web Clipping is a publishing portlet that enables you to integrate any web application with your WebCenter Portal: Framework application. Web Clipping is designed to give you quick integration by leveraging the existing user interface of a web application. With Web Clipping, you can collect web content into portlets in a single centralized web page. You can use Web Clipping to consolidate content from web sites scattered throughout a large organization.

This chapter includes the information you are required to know to use Web Clipping in your Oracle JDeveloper environment. For information about how to use Web Clipping at runtime, see the "Working with the Web Clipping Portlet" chapter in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This chapter includes the following sections:

- Section 66.1, "Introduction to Web Clipping"
- Section 66.2, "Adding Web Clipping to Your Application"
- Section 66.3, "Integrating Authenticated Web Content Using Single Sign-On"
- Section 66.4, "Advanced Features of Web Clipping"
- Section 66.5, "Current Limitations of Web Clipping"
- Section 66.6, "Troubleshooting Web Clipping"

For information about additional Web Clipping portlet producer configuration such as repository and proxy settings and producer security, see Appendix E, "Additional Portlet Configuration."

For information about the different types of portlets, producers, and other portlet technologies, see Chapter 58, "Overview of Portlets."

66.1 Introduction to Web Clipping

With Web Clipping you can clip an entire web page or a portion of it and reuse it as a portlet. You can clip basic and HTML-form-based sites. Use Web Clipping when you want to copy content from an existing web page and expose it in your Framework application as a portlet.

Web Clipping supports the following features:

- **Navigation through various styles of login mechanisms**

Web Clipping supports various login mechanisms including form- and JavaScript-based submission and HTTP Basic and Digest Authentication with cookie-based session management.

- **Fuzzy matching of clippings**

  Fuzzy matching enables the Web Clipping engine to correctly identify a web clipping and deliver it as portlet content even if the web clipping is reordered within the source page or if its character font, size, or style changes.

- **Personalization**

  Personalization enables you to expose input parameters that end users can modify when they personalize the portlet. Parameters can be exposed as public parameters that you can map as page parameters. This feature enables end users to obtain personalized clippings.

- **Integrated authenticated web content through single sign-on**

  You can use external applications and leverage Oracle Single Sign-On to clip content from authenticated external web sites.

- **Inline rendering**

  Inline rendering enables you to set up Web Clipping portlets to display links within the context of the portlet. When a user clicks a link in the Web Clipping portlet, the results display within the same portlet. You can use this feature with internal and external web sites.

- **Proxy authentication**

  Web Clipping supports proxy authentication, including global proxy authentication and authentication for each user. You can use this feature if proxy servers require authentication. You can specify proxy server authentication details, including type (Basic or Digest) and realm in the `provider.xml` file. In addition, you can specify a scheme for entering user credentials:

  – All users automatically log in using a user name and password that you provide.

  – All users are required to log in using a user name and password that they provide.

  – All public users (not authenticated into the Framework application) automatically log in using a user name and password that you provide, while valid users (authenticated into the Framework application) log in using a user name and password that they provide.

  For more information, see Section E.2.2, "HTTP or HTTPS Proxy Configuration."

- **Support for HTTPS**

  Web Clipping enables you to clip content from HTTPS-based external web sites if appropriate server certificates are acquired. For information about server certificates, see Section E.2.3.1, "Adding Certificates for Trusted Sites."

- **Open Transport API**

  By default, the Web Clipping provider supports only HTTP challenge-based authentication methods, such as Basic and Digest and form submission logins. To support custom authentication methods, such as Kerberos proxy authentication, you can use the Web Clipping Transport API. For more information, see Section 66.4.1, "Using the Web Clipping Transport API."

- **Reuse of a wide range of web content**

Web Clipping provides basic support of pages written with JavaScript, applets, and plug-in enabled content, retrieved through HTTP GET and POST methods of form submission.

Web Clipping also supports clipping of content from pages written with HTML 4.01, including:

- Clipping of `<applet>`, `<body>`, `<div>`, `<embed>`, `<img>`, `<object>`, `<ol>`, `<span>`, `<table>`, and `<ul>` tagged content
- Preservation of `<head>` styles and fonts, and CSS
- Support for UTF-8 compliant character sets
- Navigation through hyperlinks (HTTP GET), form submissions (HTTP POST), frames, and URL redirection

- **Globalization support**

  Web Clipping provides globalization support in URLs and URL parameters. For information about how Web Clipping determines the character set of clipped content, see Section 66.5, "Current Limitations of Web Clipping."

- **Persistent storage of Web Clipping definitions**

  Web Clipping definitions are stored persistently in a repository. By default, in JDeveloper, the Web Clipping producer is configured to use the file-based Oracle Metadata Service (MDS) as a repository. If you prefer, you can use a database schema for your Web Clipping repository. For information about configuring a Web Clipping repository, see Section E.2.1, "Web Clipping Repository Configuration."

- **Encryption of secure information**

  Any secure information, such as passwords, is stored in an encrypted form, according to the Data Encryption Standard (DES), using Oracle encryption technology.

66.2 Adding Web Clipping to Your Application

You can add a Web Clipping portlet to a JSP document created through JDeveloper. The Web Clipping portlet producer is included in:

- Integrated WebLogic Server (WLS), the default server installed with JDeveloper
- The `WLS_Portlets` managed server in the default domain in a full Oracle Fusion Middleware installation

To add a Web Clipping portlet to your application:

1. In JDeveloper, start Integrated WLS. For information, see Section 3.4, "Working with the Integrated WebLogic Server."
2. In JDeveloper, register the Web Clipping producer. For more information, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

   During this step, you must provide the endpoint URL, for example:

   ```
   http://localhost:7101/portalTools/webClipping/providers/webClipping
   ```

   You can choose to register the Web Clipping portlet producer from the `WLS_Portlets` managed server or Integrated WLS.

   When the producer is registered, portlets become available either in the Application Resources panel or the Resource Palette.

3. From the Application Resources panel or the Resource Palette, drag the Web Clipping portlet onto your `.jspx` page. For more information, see Section 64.3, "Adding Portlets to a Page."

   If you are using a `PanelCustomizable` or `ShowDetailFrame` component, then drag the portlet on top of that component instead of `af:form`. In the Structure pane, the Web Clipping portlet must display under the `PanelCustomizable` or `ShowDetailFrame` component. In the Page Editor, the portlet must display inside the `PanelCustomizable` or `ShowDetailFrame` component.

   Note: When you add an instance of Web Clipping to your page in JDeveloper, open the Property Inspector for the portlet and ensure that the `AllModesSharedScreen` and `RenderPortletInIFrame` properties are set as follows:

   - **AllModesSharedScreen** is set to `false` to display the Customize and Personalize in full page size. If you do not set this property to `false`, then when you edit the Web Clipping portlet at runtime, the text displayed on the page may be distorted.
   - **RenderPortletInIFrame** is set to `true` to display the Web Clipping in its own inline frame (IFRAME) in the View mode.

4. Right-click the `.jspx` page and choose **Run**.

   The Web Clipping portlet displays in your default browser.

5. Customize the Web Clipping portlet to define its contents. For more information, see the "Working with the Web Clipping Portlet" chapter in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

   Note: To clip SSL-enabled web sites, you must add certificates of those sites to the certificate store. You are not required to add certificates of SSL-enabled web sites that use Equifax, VeriSign, or Cybertrust certificates because these certificates are included in the default certificate store.

   For information about adding certificates, see Section E.2.3.1, "Adding Certificates for Trusted Sites."

66.3 Integrating Authenticated Web Content Using Single Sign-On

You can leverage Oracle Single Sign-On to integrate content from external web sites that require authentication, into a Web Clipping portlet.

Note: You can associate only one external application with a producer. For each external application, you must register a new producer. Framework application users access the authenticated content by using their user name and password for that system, not your credentials.

To integrate an external application that requires authentication:

1. Open your Framework application in JDeveloper.

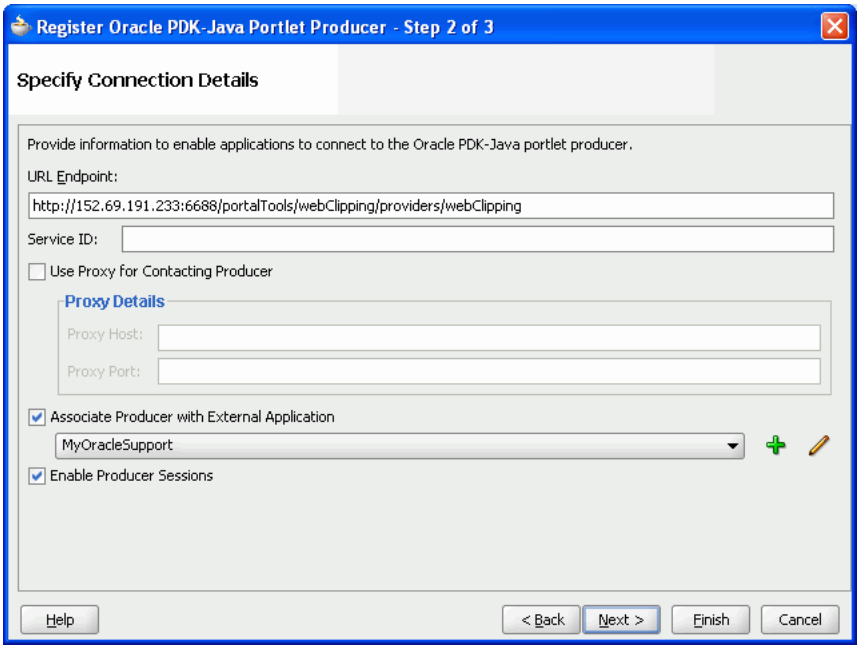
2. Register the external application, specifying the authentication information. For more information, see Section 69.13.3.2.1, "Registering an External Application in Oracle JDeveloper."

3. Register the Web Clipping producer. For information, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

   While registering the producer by using the Register Oracle PDK-Java Portlet Producer wizard, on the Specify Connection Details page, select the **Associate Producer with External Application** checkbox, and from the list of values, select the external application that you just registered. The **Enable Producer Sessions** checkbox gets selected automatically.

   Figure 66–1 shows the external application named MyOracleSupport associated with a Web Clipping producer.

Figure 66–1 Associating a Web Clipping Producer with an External Application



4. Add the Web Clipping portlet from the producer that you just registered to a `.jspx` page.

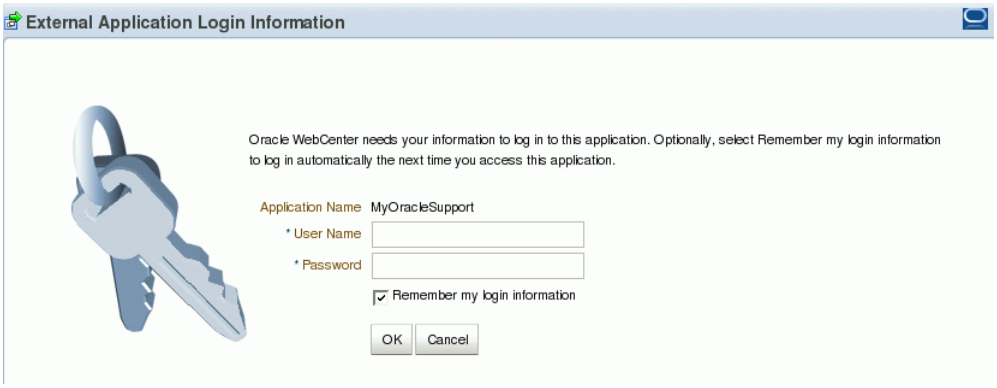
5. Right-click the `.jspx` page and choose **Run.**

6. If you did not specify shared or public credentials for the external application, then the portlet contains an **Update login information** link. Click this link.

7. Enter your credentials, and then click **OK** to log in to the external web site.

   Figure 66–2 shows the login screen of an external application named MyOracleSupport.

Figure 66–2 Logging into the Integrated External Application



8. Click the **Actions** icon on the header of the Web Clipping portlet, and select **Customize**.

   The Find a Web Clipping page displays, with the default URL for the external application displayed in the **URL Location** field (Figure 66–3).

Figure 66–3 Finding a Web Clipping



You can now select any section of a web page that you intend to display in your Web Clipping portlet. For information about how to customize a Web Clipping portlet, see the "Working with the Web Clipping Portlet" chapter in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

After you clip the required page of the external application, the web clipping, even though it is from a page requiring authentication, is available in your portlet.

66.4 Advanced Features of Web Clipping

Web Clipping supports certain advanced features. You can configure custom authentication methods by using the Web Clipping Transport API and rewrite image links to use a resource proxy.

This section includes the following subsections:

- Section 66.4.1, "Using the Web Clipping Transport API"
- Section 66.4.2, "Rewriting Image Links to Use a Resource Proxy"

66.4.1 Using the Web Clipping Transport API

To support custom authentication methods, you can use the Web Clipping Transport API. To extend the Web Clipping transport layer to support custom authentication methods, you must perform several implementation and deployment procedures.

This section includes the following subsections:

- Section 66.4.1.1, "Implementing the Web Clipping Transport API"
- Section 66.4.1.2, "Deploying the Web Clipping Transport API"

66.4.1.1 Implementing the Web Clipping Transport API

If you want to support a custom authentication method, such as Kerberos, you must first implement your own transport classes.

To implement custom transport classes:

1. Override two use cases of the `oracle.portal.wcs.transport.http.HttpTransportLiaison` interface.

   In Web Clipping, this interface is used to abstract the HTTP transport layer. By default, the two use cases of this interface are manifested by the following implementations:

   - `HttpClientStudioTransportLiaison`, which handles HTTP transport in Web Clipping Studio mode
   - `HttpClientProviderTransportLiaison`, which handles HTTP transport in Web Clipping Producer show mode

   To support more authentication methods, you must override the `addRequestHeaders` methods for both the Studio and Provider `HttpClientTransportLiaison` implementations to add your own authentication-specific headers. For more information, see the *Oracle WebLogic Server Web Clipping Transport API Reference*.

2. Compile the new subclasses and package them into a JAR file.

   For example, to compile the new subclasses, use the following command:

   ```
   javac -classpath path_to_wcejar -d classes/
   ```

   Where, *`path_to_wcejar`* refers to the path to the `wce.jar` file.

   To create the JAR file, for example, run the following command from the `classes` directory:

   ```
   jar cvf ../mytransport.jar
   ```

   Where, *`mytransport.jar`* refers to the JAR file users want to create.

66.4.1.2 Deploying the Web Clipping Transport API

After implementing the new transport classes, you must deploy the JAR file to support the custom authentication method.

To deploy the JAR file:

1. Place the JAR file into the class path or shared library that is used by the Web Clipping producer at runtime.
2. Register the transport class in the `web.xml` file for the Web Clipping producer by making the following modifications to the context parameters defined for

`HttpClientProviderTransportLiaison` and `HttpClientStudioTransportLiaison`:

- Change the parameter value for `oracle.webclipping.provider.TransportLiaisonClass` to the name of the new class extended from the `HttpClientProviderTransportLiaison` class.
- Change the parameter value for `oracle.webclipping.studio.TransportLiaisonClass` to the name of the new class extended from the `HttpClientStudioTransportLiaison` class.

3. Restart the producer server for the changes to take effect.

66.4.2 Rewriting Image Links to Use a Resource Proxy

Web Clipping enables you to rewrite image links to use a resource proxy. To enable this feature, you must add the following entry to the `web.xml` file of the Web Clipping producer:

```
<env-entry>
    <env-entry-name>oracle/webclipping/rewriteImageLink</env-entry-name>
    <env-entry-type>java.lang.Boolean</env-entry-type>
    <env-entry-value>false</env-entry-value>
</env-entry>
```

66.5 Current Limitations of Web Clipping

When you use Web Clipping, you should be aware of the following limitations.

- If the site that you intend to clip uses a large amount of JavaScript to manipulate cookies or uses the `document.write` JavaScript method to modify the HTML document being written, then you may not be able to clip content from the site.
- When you integrate with partner applications (by using `mod_osso`), you cannot clip directly through those partner applications in an authenticated manner. However, you can use partner applications through the external application framework.
- You cannot use the Web Clipping portlet to clip Oracle Portal pages and ADF pages. As a workaround, reregister the same producer in the destination portal and edit the portal manually.
- You cannot use the Web Clipping portlet to clip a web page that contains multiple frames, that is, a frameset.
- Note the following about Web Clipping and the use of a CSS:
  - If a web page contains multiple portlets that use a CSS, then they should not conflict if the CSS uses distinct style names, such as `OraRef`, to specify a style within an HTML tag, rather than using an HTML tag name, such as `<A>`, as the name of the style.
  - If one portlet uses a CSS, and that CSS overwrites the behavior of HTML tags by using the name of the tag, such as `<A>`, as the name of the style, and a second portlet on the same page does not use a CSS, the second portlet is affected by the style instructions of the CSS of the first portlet.
  - If two portlets on the same page use a different CSS and each CSS overwrites the behavior of HTML tags by using the name of an HTML tag, such as `<A>`,

as the name of the style, then the style that is displayed depends on the browser.

- Web Clipping checks for globalization support settings in the following way:
  1. Web Clipping checks the `Content-Type` in the HTTP header for the `charset` attribute. If this is present, then it assumes that this is the character encoding of the HTML page.
  2. If the `charset` attribute is not present, then Web Clipping checks the HTML `META` tag on the page to determine the character encoding.
  3. If the HTML `META` tag is not found, then Web Clipping uses the `charset` in the previous browsed page. If this is the first page, then it defaults to the ISO-8859-1 character encoding.
  4. If the value of the `charset` for `Content-Type` or `META` tag is not supported (for example, if the `charset` was specified as `NONE`), then Web Clipping uses the default character set, ISO-8859-1, not the `charset` in the previously browsed page.
- To use the Web Clipping portlet, you must use Netscape 7.0 or later, Microsoft Internet Explorer 5.5 or later for Windows 2000, or Microsoft Internet Explorer 6.0 or later for Windows XP. If you use browser versions older than these, then you may encounter JavaScript errors.

66.6 Troubleshooting Web Clipping

This section provides information to help you troubleshoot problems you may encounter while using Web Clipping.

Encountered "x" at line n, column n. Was expecting one of: "x", "y" ...

Parser error message written to the log file.

Problem

The web content displayed in the Web Clipping portlet contains invalid HTML or JavaScript.

Solution

This is a site-specific issue, not a Web Clipping error. Contact the site's administrator for assistance.

Part XI

Delivering Personalized Content

Part XI contains the following chapter:

- Chapter 67, "Personalizing WebCenter Portal Applications"
- Chapter 68, "Using Personalization APIs Externally"

67

Personalizing WebCenter Portal Applications

Personalization for WebCenter Portal delivers targeted content based on both user and application context. WebCenter Portal's Personalization also provides a run-time system and associated tools that allow the declarative definition of application flow.

This chapter describes how to integrate Personalization into a WebCenter Portal: Framework application and includes the following sections:

- Section 67.1, "Introduction to Personalization"
- Section 67.2, "Integrating Personalization in Your Application"
- Section 67.3, "Tutorial: Creating, Testing and Deploying a Simple Application"
- Section 67.4, "Extending Personalization"

For more information about Personalization, see:

- Appendix G.9, "ELs Related to the Personalization Service"
- "Managing the Personalization Service" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*

67.1 Introduction to Personalization

Personalization provides a dynamically derived user experience for your WebCenter Portal application. Personalization evaluates defined sources of input data, generates a decision based on that evaluation, and applies this information to a declaratively defined Personalization scenario. Personalization, for example, can return content or change application flow based on information about a user in a Human Resources database, targeting the application experience for that specific user.

This section provides an overview of the features and requirements of Personalization, and contains the following subsections:

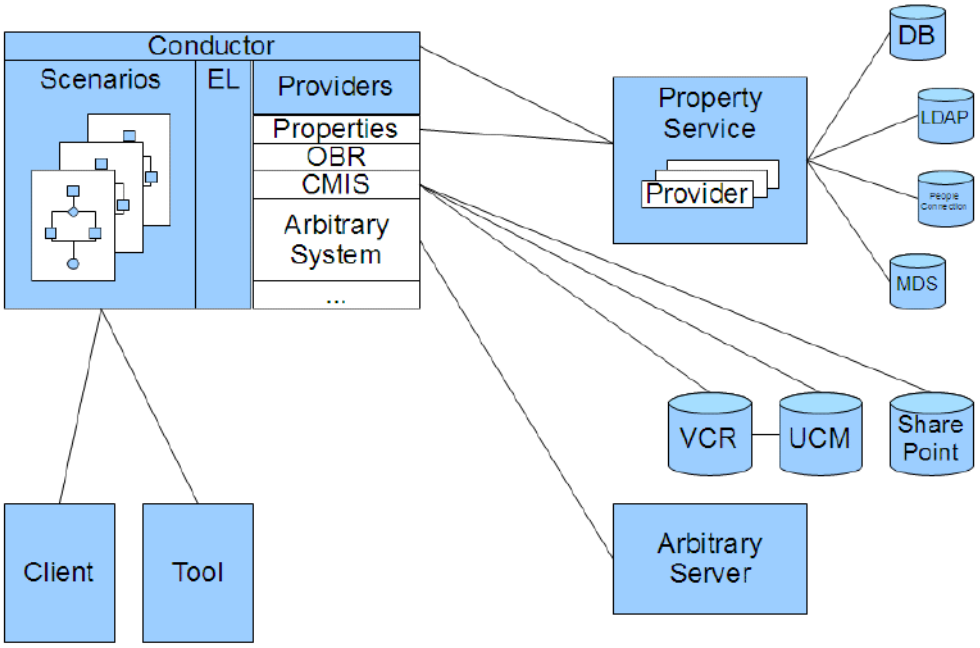
- Section 67.1.1, "Personalization Architecture"
- Section 67.1.2, "Personalization Conductor"
- Section 67.1.3, "Personalization Property Service"
- Section 67.1.4, "Personalization Providers"

67.1.1 Personalization Architecture

Figure 67–1 shows the architecture and out-of-the box services and components that comprise Personalization. These and other Personalization components are described in:

- Section 67.1.2, "Personalization Conductor"
- Section 67.1.3, "Personalization Property Service"
- Section 67.1.4, "Personalization Providers"

Figure 67–1 Personalization Services Architecture



67.1.2 Personalization Conductor

This section describes the role of the *Personalization Conductor*. The Conductor is the heart of the Personalization engine, and contains and runs the units of work called *scenarios*. The Conductor runs the prepared scenarios, and provides access to provider extensions that plug into the Conductor. These aspects of the Conductor are described in the following sections:

- Section 67.1.2.1, "Personalized Scenarios"
- Section 67.1.2.2, "Personalized Application Flow"
- Section 67.1.2.3, "Property Set Integration"
- Section 67.1.2.4, "Provider Integration"

67.1.2.1 Personalized Scenarios

A *scenario* is defined by a script you create using the Scenario Editor in JDeveloper (or provide as an XML file) based on a simple syntax that allows conditional statement evaluation resulting in the return (to the client) of some form of content. Based on function provider expressions, a script action may then return filtered content to the client application using a Data Provider and a query. A scenario, rather than returning

targeted content, can also be used to make dynamic changes to application flow, resulting in a personalized user experience within the application.

In JDeveloper, you can use the Scenario Editor to graphically design your scenario. You can also create XML file-based scenarios. See Section 67.2.2.4, "Specifying Scenario Flow Using Node Types" and Section 67.2.3, "Creating File-Based Scenarios" for more information about creating scenarios.

67.1.2.2 Personalized Application Flow

As well as returning content targeted to a specific user, scenarios executing within the Conductor can also control application flow, render user interface components, and execute application-related events. Using the Conductor's public Java and Expression Language (EL) API, you can provide seamless integration with an existing Java or Web-based application. See Chapter 68, "Using Personalization APIs Externally" for more information about using Java and EL APIs.

67.1.2.3 Property Set Integration

The Property Service uses *property sets* to organize sets of user or application data, and *property definitions* to assign a data type to property data. You can extend the Property Service to access additional user profile data in an enterprise LDAP repository, or to access additional repositories such as Oracle MDS. For more information about the Property Service and property sets, see Section 67.1.3, "Personalization Property Service."

67.1.2.4 Provider Integration

Providers provide a way to access external resources within your scenario. As well as three out-of-the-box providers for the Property Service, Content Server Provider (CMIS), and Activity Graph, the Conductor also supports an extensible architecture that lets you implement and access custom providers from within your Scenarios. For more information about providers, see Section 67.1.4, "Personalization Providers."

67.1.3 Personalization Property Service

This section describes the *Personalization Property Service*. The Property Service provides a simple yet scalable way for developers to access user data.

This section includes the following subsections:

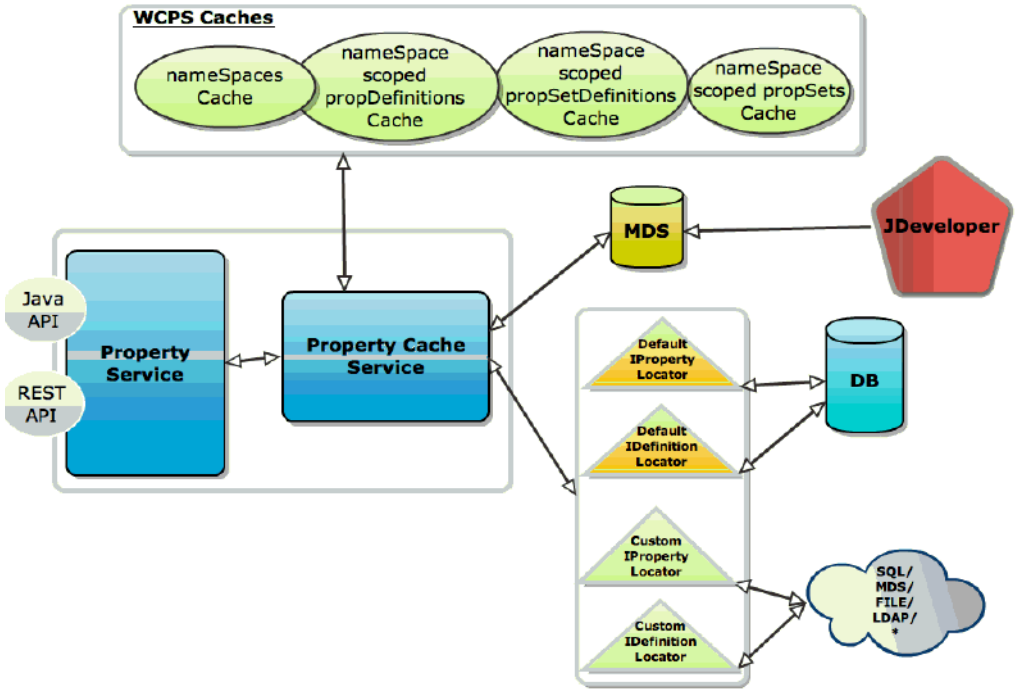
- Section 67.1.3.1, "Property Service Architecture"
- Section 67.1.3.2, "Property Sets"
- Section 67.1.3.3, "Property Locators"

67.1.3.1 Property Service Architecture

The Property Service uses Java and REST APIs to store and retrieve properties about a user, such as their age or gender, or other information such as the current time. These properties are part of a *property set*. A property set can have a schema defined (the schema defines types for properties), or it can be created on demand without a schema. A property set can have properties that are retrieved from different locations such as the Identity Store LDAP server, a file, or using a SQL query. For more information about using Java and REST APIs, see Chapter 68, "Using Personalization APIs Externally"

The Property Service uses the default Personalization cache (or Coherence cache if available) and by default stores property sets in a database. The Property Service is also extensible allowing additional repositories such as Oracle MDS to be configured.

Figure 67–2 Property Service Architecture



67.1.3.2 Property Sets

The Property Service uses *property sets* to organize sets of user or application data, and *property definitions* to assign a data type to property data. The existing user profile data in an enterprise LDAP repository can also be made accessible by extending the Property Service. See Section 67.2.2.5, "Defining Property Sets and Property Locators" for more information about the property sets. In a JDeveloper project for a WebCenter Portal application, the PropertySet Editor creates property namespace files containing `namespace`, `PropertySet`, and `Property` definitions.

67.1.3.3 Property Locators

A property locator is invoked by the Property Service to access external stores of properties. For example, Personalization provides an out-of-the-box People Connections property locator that uses the People Connections REST service to access profile information for users. For more information about property locators, see Section 67.2.2.5, "Defining Property Sets and Property Locators." For more information about the People Connections property locator, see Section 67.1.4.1.4, "People Connections Locator."

67.1.4 Personalization Providers

A provider is an interface point for the Conductor that lets it communicate with external services. Personalization provides out-of-the-box providers for the Property Service, Oracle WebCenter Content: Content Server, and Activity Graph, and a locator used by the Property Service provider to allow integration with People Connections.

There are also two types of custom providers you can implement: *data providers* and *function providers*.

This section describes the out-of-the-box providers and the two types of custom providers in the following subsections:

- Section 67.1.4.1, "Out-of-the-Box Providers"
- Section 67.1.4.2, "Custom Data Providers"
- Section 67.1.4.3, "Function Providers"

67.1.4.1 Out-of-the-Box Providers

This section describes the out-of-the-box providers you can access from within your scenarios, and a locator used by the Property Service provider to allow integration with People Connections.

This section contains the following subsections:

- Section 67.1.4.1.1, "Property Service Provider"
- Section 67.1.4.1.2, "CMIS Provider"
- Section 67.1.4.1.3, "Activity Graph Provider"
- Section 67.1.4.1.4, "People Connections Locator"

67.1.4.1.1 Property Service Provider

The Property Service Provider allows integration with the Property Service. This provider allows you to retrieve property sets for a particular property set definition, retrieve a property set by name, or return a specific property from a property set. For more information about the Property Service Provider, see Section 67.2.2.8.1, "Using the Property Service Provider."

67.1.4.1.2 CMIS Provider

The CMIS Provider provides services to retrieve and search for content from standards-based CMIS (Content Management Interoperability Services) content servers, specifically provided by the Oracle WebCenter Content Server. The CMIS Provider also has utility function providers to convert results to simplified form. For information about integrating the CMIS Provider, see Section 67.2.2.8.2, "Using the CMIS Provider." For information about creating and editing connection configuration for the Content Server provider, see "Configuring the REST Service Identity Asserter" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

67.1.4.1.3 Activity Graph Provider

The Activity Graph Provider (ActivityGraphProvider) provides integration with the Activity Graph engine (the basis for the Activity Graph service), which provides recommendations about what a user may be interested in connecting with based on analytics within WebCenter. For example, the Activity Graph Provider method `recommendedUsers`, queries the Activity Graph engine for a list of users that a given user might want to connect with.

The Activity Graph engine provides a central repository for actions that are collected by enterprise applications. For more information about the Activity Graph service and the Activity Graph engine, see Section 46, "Integrating the Activity Graph Service."

The Activity Graph provider is built on two REST services for Activity Graph running on the WebCenter Portal: Spaces server, and consequently requires that WebCenter Portal: Spaces be installed in your environment. For more information about

integrating the Activity Graph provider in your scenario, see Section 67.2.2.8.3, "Using the Activity Graph Provider." For information about creating and editing connection configuration for the Activity Graph provider, see "Configuring the Activity Graph Provider" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

67.1.4.1.4 People Connections Locator

Although The People Connections locator is an ILocator used by the Property Provider to interface with the WebCenter People Connections service. One of the benefits of the People Connections service is that it can access user profile information stored on WebCenter Portal: Spaces. For more information about the People Connections service, see Part VI, "Working with the People Connections Service". For more information about the People Connections locator, see Section 67.2.2.8.1, "Using the Property Service Provider." For information about creating and editing connection configuration for the People Connections locator, see "Configuring the Oracle People Connections Locator" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

67.1.4.2 Custom Data Providers

A *data provider* is a component that plugs into the Conductor architecture and acts as a delegate for some service. Data returned from a data provider can be used as-is (Java classes), or chained to be used as input to other data providers in the same scenario.

The Property Service Provider, Activity Graph Provider, and the CMIS Provider are all examples of data providers that are available out-of-the-box. But data providers (`oracle.wcps.conductor.provider.IDataProvider`) also provide an extension point for the Conductor so that you can write a custom data provider to, for example, integrate with legacy systems or external data stores. You can then call your custom (or out-of-the-box) data provider using the `Invoke Provider` node from within a scenario.

Data providers typically operate on the input context of a given user or an external application data set (or both). Personalization includes a Property Service that can provide typed data storage for user or application data, and is accessible as input to providers. For more information about creating your own data provider, see Section 67.4.1, "Adding Custom Providers."

67.1.4.3 Function Providers

Function providers (`oracle.wcps.conductor.provider.IFunctionProvider`) also provide an extension point for the Conductor to create utility methods (for example, for data manipulation or transformation, or business rule calculation) that can be invoked using Expression Language (EL) expressions. A simple example might be: `${strings:concat('string1','string2')}`). The CMIS and Activity Graph providers each have their own function provider to facilitate data transformations, among other utilities. For more information about Personalization ELs, see Appendix G.9, "ELs Related to the Personalization Service."

67.2 Integrating Personalization in Your Application

This section describes how to integrate Personalization in your Framework application.

This section contains the following subsections:

- Section 67.2.1, "Personalization Requirements"

- Section 67.2.2, "Authoring Personalized Scenarios in JDeveloper"
- Section 67.2.4, "Displaying Targeted Content at Runtime"

67.2.1 Personalization Requirements

This section describes the design-time system requirements and dependencies for developing WebCenter Portal applications with JDeveloper that integrate Personalization services. For a complete list of requirements, dependencies, and options for Personalization, see "Personalization Prerequisites" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

- WebCenter Portal: Spaces must be installed if the CMIS Provider, Activity Graph Provider, or People Connections locator are being used in a scenario. Note that for basic iterative development WebCenter Portal: Spaces is not required to be installed on your domain.
- Before you can use REST APIs, you must take the steps outlined in "Before You Begin: Performing Required Configurations" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- Personalization relies on WebCenter Trust Services to provide single sign-on (SSO) between different managed servers within the WebCenter domain, including the Oracle WebCenter Content: Content Server that is accessed through the CMIS Provider. For JDeveloper's integrated domain, the Trust Services must be configured using a WLST script (`configureWCPS.py`) located in the `DefaultDomain/scrpts-wcps` directory. For more information about configuring the WebCenter Trust Services and single sign-on using this script see "Configuring Single Sign-on" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.
- The REST service must be configured. For information about how to configure the REST service, see the chapter "Managing REST Services" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal.

Note: Although you can optionally include the out-of-the-box providers for Activity Graph and People Connections in your application, you will not be able to test results in JDeveloper. Note also that these providers are only partially configured, and you must complete their configuration as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* in the sections on "Configuring the Oracle People Connections Service" and "Configuring the Activity Graph Service."

67.2.2 Authoring Personalized Scenarios in JDeveloper

This section describes how to author personalized scenarios using the Personalization tools in JDeveloper, and includes the following subsections:

- Section 67.2.2.1, "Creating a Properties Namespace"
- Section 67.2.2.2, "Creating Property Sets and Property Definitions"
- Section 67.2.2.3, "Creating a New Scenario in the Scenario Editor"
- Section 67.2.2.4, "Specifying Scenario Flow Using Node Types"
- Section 67.2.2.5, "Defining Property Sets and Property Locators"
- Section 67.2.2.6, "Implementing and Calling Function Providers"

- Section 67.2.2.7, "Implementing and Calling Data Providers"
- Section 67.2.2.8, "Using the Out-of-the-Box Providers"

67.2.2.1 Creating a Properties Namespace

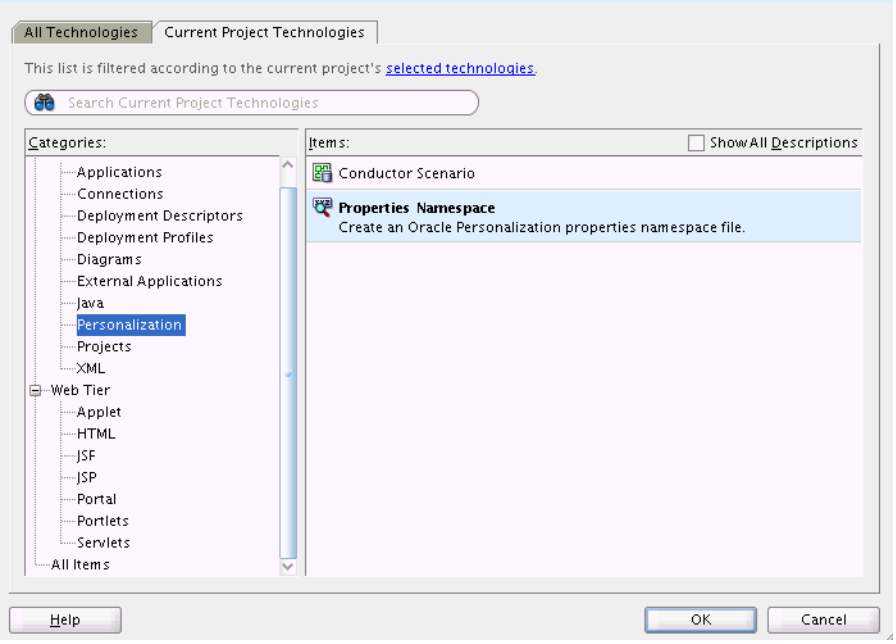
The Properties Namespace file contains the property definitions and property set definitions you define.

To create a Properties Namespace file in JDeveloper:

1. From within your application in JDeveloper, right-click the application in the navigation bar.

   The New Gallery dialog displays (see Figure 67–3).

Figure 67–3 New Gallery Dialog

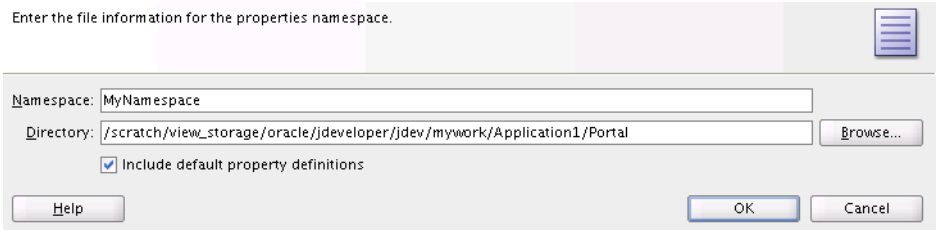


> **Note:** If the Personalization category doesn't display, open the All Technologies tab and select it, or click **selected technologies** and add the Personalization technology to the project.

2. Choose **Properties Namespace** and click **OK**.

   The Create Namespace dialog displays (see Figure 67–4).

Figure 67–4 Create Namespace Dialog



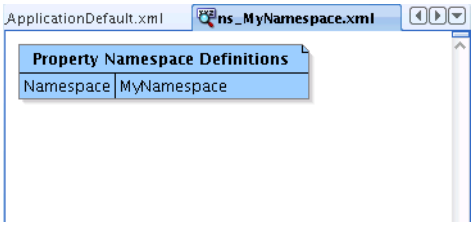
3. Enter a name for the namespace, select the **Include default property definitions** checkbox to add initial default property definitions to the namespace, and click **OK**.

   The **Include default property definitions** checkbox controls whether initial default property definitions are added to the namespace file. These are: `IntDef`, `NumberDef`, `StringDef`, `BooleanDef`, `DateTimeDef`, `ClobDef`, `IntArrayDef`, `NumberArrayDef`, `StringArrayDef`, `BooleanArrayDef`, `DateTimeArrayDef`, `ClobArrayDef`.

   The property definitions can be removed, if needed afterwards. Or, you can leave the checkbox unselected to start with no initial property definitions and create them as needed in the Scenario Editor.

   A graphic representation of the namespace file displays in the Design pane (see Figure 67–5).

Figure 67–5 Design Pane - Namespace Definition



4. Continue by defining the property sets and properties that you will need in your scenario as described in Section 67.2.2.2, "Creating Property Sets and Property Definitions."

67.2.2.2 Creating Property Sets and Property Definitions

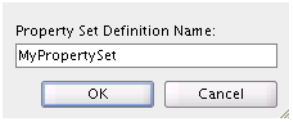
Properties are the bits of data about someone or something on which, for example, you can base evaluations of conditionals in your scenario's flow. A property, for example, could contain a person's age or gender. A property set is simply a collection or container of related properties.

To define a property set and associated properties:

1. From within your application in JDeveloper, right-click the namespace in the Design pane, select **Create New Property Set Definition**, and click **OK**.

   The Create New Property Set dialog displays (see Figure 67–6).

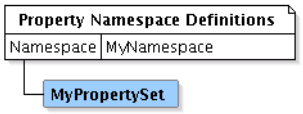
Figure 67–6 Create New Property Set Dialog



2. Enter a name for the property set and click **OK**.

   A node for the new property set displays in the Design pane (see Figure 67–7).

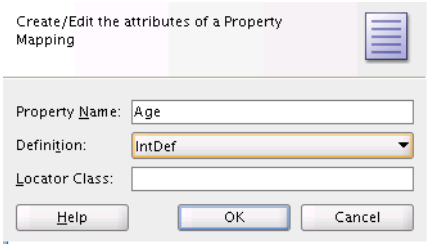
Figure 67–7 Design Pane - New Property Set Definition



3. To add properties to the new property set, right-click the property set, and select **Create New Property Definition Mapping**.

   The Create Property Definition dialog displays (see Figure 67–8).

Figure 67–8 Create Property Definition

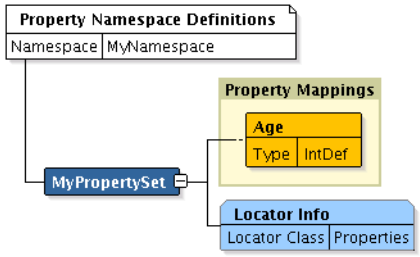


4. Enter a **Property Name** and **Definition** (the value type), and click **OK**.

   To create a new property definition, select `New Definition` in the **Definition** drop down list.

   The new property appears as part of the property set's graphic representation (see Figure 67–9).

Figure 67–9 Design Pane - Property Definition



5. Add any additional properties required to the property set (or create new property sets) and then continue by creating the scenario as described in Section 67.2.2.3, "Creating a New Scenario in the Scenario Editor."

67.2.2.3 Creating a New Scenario in the Scenario Editor

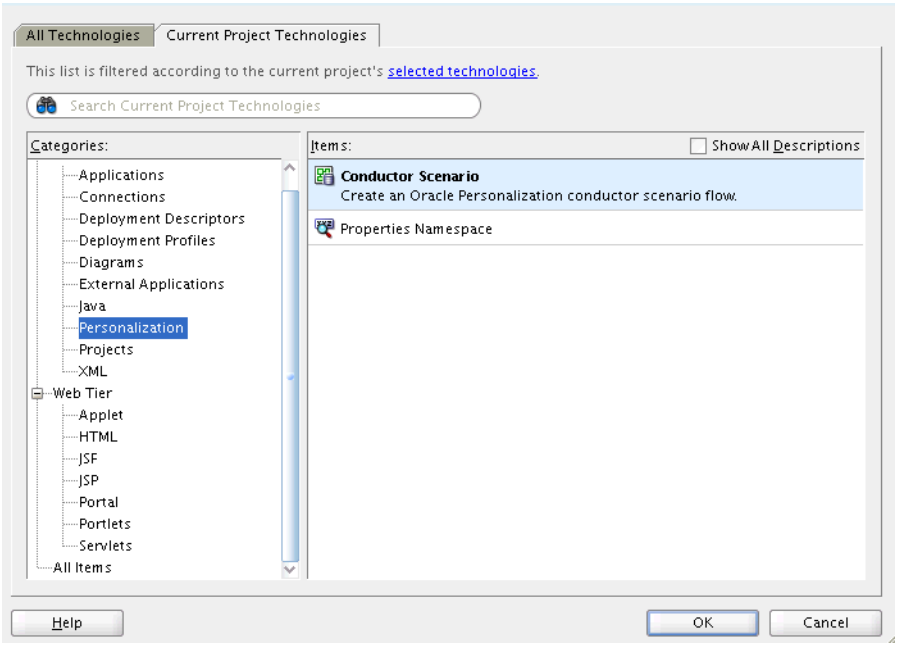
The Conductor's Scenario Editor provides a visual tool for viewing and editing scenario definitions managed by the Conductor. The Scenario Editor stores scenarios in both file and visual formats and is available from within JDeveloper. Before creating a scenario, you should have created a namespace and at least some of the property sets and properties you need for the scenario.

To create a new scenario using the Scenario Editor in JDeveloper:

1. From within your application in JDeveloper, right-click the application in the navigation bar.

   The New Gallery dialog displays (see Figure 67–10).

Figure 67–10 New Gallery Dialog



> **Note:** If the Personalization category doesn't display, open the All Technologies tab and select it, or click **selected technologies** and add the Personalization technology to the project.

2. Select **Conductor Scenario** and click **OK**.

   The Scenario Definition dialog displays (see Figure 67–11).

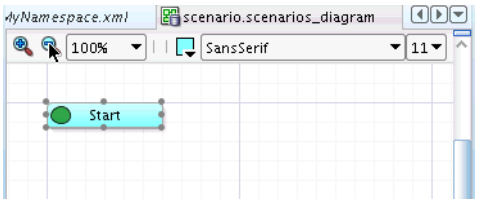
Figure 67–11 Scenario Definition Dialog



3. Select **Create a new conductor scenario file** and modify the file name and where the file is stored if necessary. Note that the scenario name's extension (`.scenarios_diagram`) cannot be changed.

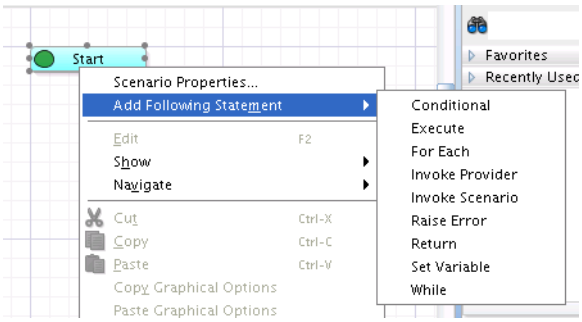
   The Scenario Editor displays with a new Start node (see Figure 67–12).

Figure 67–12 Scenario Editor - Start Node



4. To add nodes to the scenario, right-click the node to which you want to add a new node, select **Add Following Statement**, and chose the node type from the list as shown in Figure 67–13.

Figure 67–13 Adding a Node in the Scenario Editor



Right-click the new node to expose options for setting node properties and extending the flow. For a description of available node types, see Section 67.2.2.4, "Specifying Scenario Flow Using Node Types."

67.2.2.4 Specifying Scenario Flow Using Node Types

The Scenario Editor uses a tree-structured fixed layout where the flow is represented by a combination of subtrees (for decisions) and internal label references to represent flow for loops and complex nested if/else decisions.

Node Types

- **Start:** The Start node is a root level node that defines where a scenario begins.
- **Return:** The Return node halts execution of the scenario, and returns the specified results of evaluation of an EL expression to the caller.
- **Variable Assignment:** The variable assignment node lets you define a variable that is scoped within the context of the currently running scenario and initialize it with an expression.
- **Execute:** The Execute node invokes a specified EL expression with no expected return value. Similar to Variable Assignment, only the results of the expression are not stored within the Scenario context.
- **Condition/Otherwise:** The Condition node evaluates an EL expression and executes the contained statements if the EL expression evaluate to true. Statements within the Otherwise block will only execute if all condition statements evaluate to false.
- **For Each:** For Each nodes are a matched pair of nodes that provide looping/iterating functionality over a collection of things.

- **While:** Provides looping/iterating functionality as long as a specified EL expression evaluates to true.
- **Raise Error:** Raise and throw an error with the specified error message
- **Invoke Provider:** Invokes a named implementation of IDataProvider. Results are stored within the context of the currently running Scenario.
- **Invoke Scenario:** Invokes another named Scenario within the same namespace. Results are stored within the context of the currently running Scenario

67.2.2.5 Defining Property Sets and Property Locators

The Property Editor lets you view and edit the property set definitions and their associated property definitions managed by the Property Service. The Property Editor communicates and persists data through the Property Service.

Property sets and properties within a property set can be stored in many different locations and formats, so a locator facility is needed to retrieve them. A default property set locator is provided that retrieves and stores property set data to a pre-configured database. This default locator is used when no other is specified for the property set.

A custom locator for the People Connections service is also provided out-of-the-box (for more information, see Section 67.1.4.1.4, "People Connections Locator"), but you may want to define additional locators for other data sources. You can define multiple locators for properties and property sets, and they can also be defined in order of precedence. For step-by-step instructions for how to implement a locator see Section 67.4.2, "Adding Custom Locators."

67.2.2.6 Implementing and Calling Function Providers

The `IFunctionProvider` is a marker interface that allows utility code to be accessible from the Expression Language services within the Conductor. Implementations of `IFunctionProvider` contain annotated public and static methods. Example 67–1 shows an example function provider:

Example 67–1 Example Function Provider

```
@FunctionProvider
(
    prefix="interpreter",
    resourceBundle="sample.InterpreterResources",
    nameBundleKey="provider.name",
    descriptionBundleKey="provider.description"
)
public class InterpreterProvider
implements IFunctionProvider
{

    @PublicFunction
    (
        functionName="english",
        descriptionBundleKey="english.description"
    )
    public static String
        interpretEnglish(@PublicParameter(parameterName="phrase",

descriptionBundleKey="phrase.description")String phrase)
    {
        String result = "Not Understood";
```

```
        if ("Bonjour".equals(phrase))
        {
           result = "Hello";
        }

        return result;
    }

    @PublicFunction
    (
        functionName="french",
        descriptionBundleKey="french.description"
    )
    public static String
        interpretFrench(@PublicParameter(parameterName="phrase",

descriptionBundleKey="phrase.description")String phrase)
    {
        String result = "Non compris";

        if ("Hello".equals(phrase))
        {
           result = "Bonjour";
        }

        return result;
    }
}
```

You would call the function provider in Example 67–1 using the following expression syntax:

```
${interpreter:english("Bonjour")}
${interpreter:french("Hello")}
```

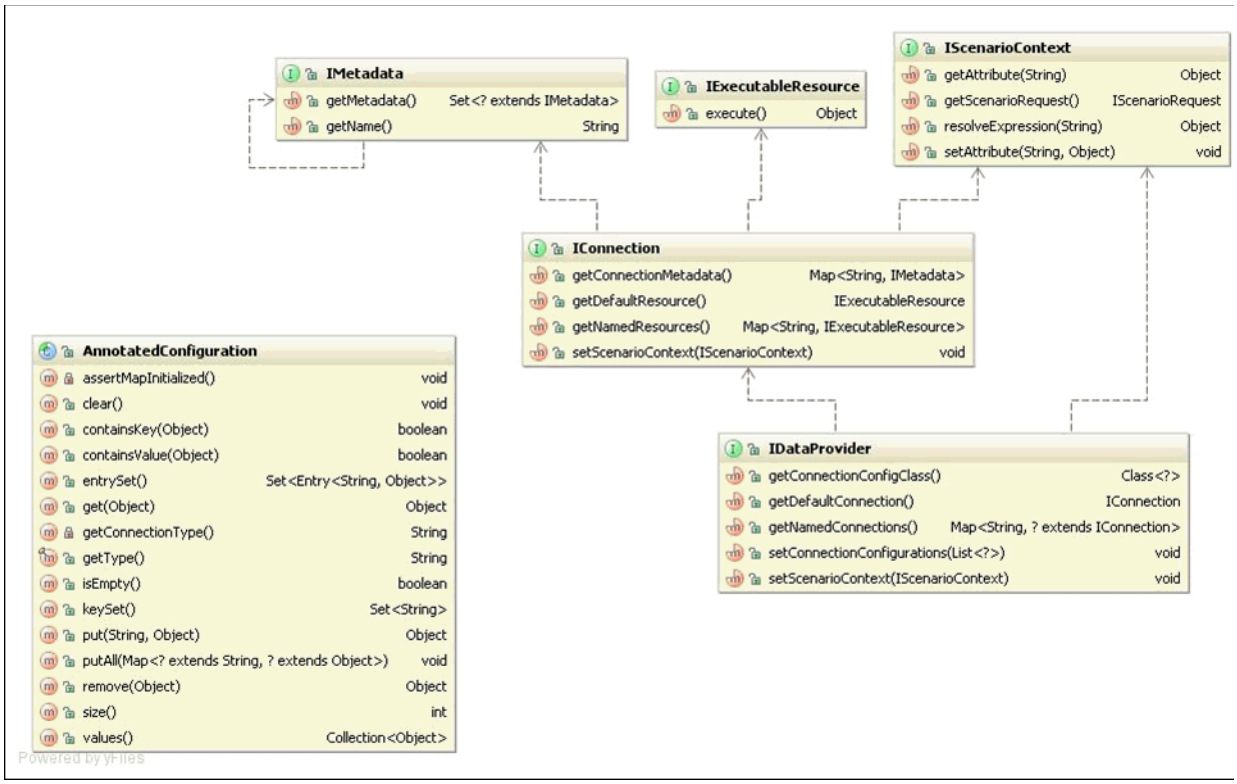
Here is an example of how to assign a variable to the results of a function provider based on Example 67–1:

```
<assign-variable>
     <variable>greetings</variable>
    <expression>${interpreter:english("Bonjour")}</expression>
</assign-variable>
```

67.2.2.7 Implementing and Calling Data Providers

Data provider implementations are a more structured way of integrating custom functionality with scenario execution. Unlike function providers, data providers have access to a connection configuration framework built into the Conductor so that environment-specific configuration data can be used. The following diagram illustrates the class interaction for data providers:

Figure 67–14 Class Interaction for Data Providers



IDataProvider

Data providers are responsible for returning abstract named connection implementations of `IConnection`. These named connections are usually backed by environmental configurations provided by the Conductor's connection configuration service, but do not have to be. Should the data provider happen to use the configuration service, the implementation must return a class that extends `oracle.wcps.connection.configuration.AnnotatedConfiguration` from the `getConnectionConfigClass()` method. This class must contain public annotated fields that represent configuration data specific to the data provider.

IConnection

Implementations of `oracle.wcps.conductor.provider.IConnection` are responsible for returning (named and default) Executable Resources. Each instance of `IConnection` can represent a single connection configuration and should be constructed within the instance of `IDataProvider` (not static).

IExecutableResource

Implementations of `oracle.wcps.conductor.provider.IExecutableResource` contain most of the integration code with the external system and the instance should be constructed within the instance of `IConnection` (not static). The default method from within `IExecutableResource` is the `execute` method that takes no arguments, but custom annotated methods can be written within the implementation if `IExecutableResource` and exposed to scenario execution. For example, the following custom method can be invoked from a scenario:

```
@PublicFunction
            (
```

```
                functionName="custom",
                descriptionBundleKey="customFunction.description"
        )
    public String callCustomMethod(@PublicParameter(parameterName="customParam",
        descriptionBundleKey="customParam.description") String param)
    {
        return param + " Results";
    }
```

Here is how the provider with the custom method is invoked from within the scenario:

```
    <call-provider>
        <provider-name>myCustomProvider</provider-name>
        <connection-name>${connectionName}</connection-name>
        <method-name>custom</method-name>
        <variable>searchResults</variable>
        <input>
            <parameter>
                <common:name>customParam</common:name>
                <common:value>${someScenarioVariable}</common:value>
            </parameter>
        </input>
    </call-provider>
```

67.2.2.8 Using the Out-of-the-Box Providers

Personalization provides three providers out-of-the-box providers for use within scenarios: the Property Service Provider, the Content Server Provider (CMIS), and Activity Graph. This section describes how you can integrate these providers in your application in the following subsections:

- Section 67.2.2.8.1, "Using the Property Service Provider"
- Section 67.2.2.8.2, "Using the CMIS Provider"
- Section 67.2.2.8.3, "Using the Activity Graph Provider"

67.2.2.8.1 Using the Property Service Provider

The Property Service Provider (`oracle.PropertiesServiceProvider`) allows integration with the Personalization Server Property Service. This provider lets you to retrieve property sets for a particular property set definition, retrieve a property set by name, or return a specific property from a property set. If you are using the People Connections locator to integrate profile data into your scenario, then you must configure the Property Service before using it. For more information about the People Connections locator, see Section 67.1.4.1.4, "People Connections Locator." For information about configuring the Property Service provider, see "Configuring the Oracle People Connections Locator" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

67.2.2.8.2 Using the CMIS Provider

The CMIS provider (`CMISProvider`) provides services to search for and retrieve content from standards-based CMIS (Content Management Interoperability Services) content servers. For WebCenter, this is specifically provided by the Oracle WebCenter Content: Content Server. The CMIS provider also has a utility function provider (`CMISFunctionProvider`) to convert results to simplified form. Before trying the examples below, be sure to configure your CMIS provider connection as shown in the section on "Configuring the CMIS Provider" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal.

Example CMIS Provider Scenario

Create a scenario using the following (remember to set the `content-type` of `application/xml`):

Request URI:

```
POST http://localhost:8891/wcps/api/conductor/namespaces/myNameSpace/scenarios
```

Request Body:

```
<scenarioMetadata>
<scenario xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <body>
    <call-provider>
      <provider-name>CMISProvider</provider-name>
      <connection-name>CmisConfigConnection</connection-name>
      <variable>exprString</variable>
      <input>
        <parameter>
            <common:name>search</common:name>
            <common:value>SELECT * FROM cmis:document WHERE cmis:name LIKE
'flowers'</common:value>
        </parameter>
      </input>
    </call-provider>
    <return>
      <expression>${exprString}</expression>
    </return>
  </body>
  <name>CMIS Scenario</name>
  <comments>CMIS Scenario comments</comments>
  <tags>
    <tag>CMIS Scenario tag</tag>
  </tags>
</scenario>
</scenarioMetadata>
```

Executing the Scenario

Request URI:

```
POST
http://localhost:8891/wcps/api/conductor/namespaces/myNameSpace/scenarios/CMIS%20S
cenario
```

Request Body:

```
<parameters/>
```

Using the CMISFunctionProvider

A function provider is a utility class that can be invoked from within a scenario. The `CMISFunctionProvider` supports several methods (see the JavaDoc for the class for more information). Here are two example methods:

```
/**
     * Convert the array of input IDs to a CMIS query in the form of 'IN'
     * @param ids Array of document IDs, in the form of dDocName only
     * @return CMIS query String
     */
public static String getCMISQueryForDocIDs(
```

```
@PublicParameter(parameterName="repository",descriptionBundleKey="getCMISQueryForD
ocIDs.parameter.description.inputList") String repository,

@PublicParameter(parameterName="ids",descriptionBundleKey="getCMISQueryForDocIDs.p
arameter.description.inputList") List<String> ids))
...


/**
     * Convert the array of input IDs to a CMIS query in the form of 'IN'
     * @param ids, in the form of a WebCenter document ObjectURN.  For example:
stanl18-ucm11g#dDocName:MOUNTAINS
     * @return CMIS query String
     */
 public static String
    getCMISQueryForDocIDsFromFullID(@PublicParameter(parameterName="ids",

descriptionBundleKey="getCMISQueryForDocIDs.parameter.description.inputList")
List<String> ids)
```

Using the CMISFunctionProvider in a Scenario

The following scenario snippet shows how data might be fed into the `CMISFunctionProvider` to create a CMIS query for multiple nodes, where the query is then sent to the CMIS provider. (In a more realistic case, rather than generate the data, we could have passed the results from an Activity Graph provider call, for example.)

```
<scenario:scenario
xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"

xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <comments>testFunctionProviderScenario4 comments</comments>
  <body>
   <assign-variable>
      <variable>ids</variable>
      <expression>${collections:new()}</expression>
    </assign-variable>
    <execute>

<expression>${collections:append(ids,'StellentRepository#dDocName:MOUNTAINS')}</ex
pression>
    </execute>
    <execute>

<expression>${collections:append(ids,'StellentRepository#dDocName:PARKS')}</expres
sion>
    </execute>
    <execute>

<expression>${collections:append(ids,'StellentRepository#dDocName:PATAGONIAPARKS')
}</expression>
    </execute>

    <!-- Assign those as IDs to send to CMIS -->
    <assign-variable>
        <variable>cmisQuery</variable>

<expression>${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}</expression>
    </assign-variable>
```

```
    <!-- Make the call to CMIS -->
    <call-provider>
      <provider-name>CMISProvider</provider-name>
      <connection-name>NonProxyConnection</connection-name>
      <variable>cmisResults</variable>
      <input>
        <parameter>
            <common:name>search</common:name>
            <common:value>${cmisQuery}</common:value>
        </parameter>
      </input>
    </call-provider>
    <return>
      <expression>${cmisResults}</expression>
    </return>
  </body>
  <name>testFunctionProviderScenario4</name>

</scenario:scenario>
```

67.2.2.8.3 Using the Activity Graph Provider

The Activity Graph Provider (`ActivityGraphProvider`) provides integration with the Activity Graph service, which provides recommendations based on analytics within WebCenter. Before attempting the examples below, be sure to configure the connection for the Activity Graph service as shown in the section on "Configuring the Activity Graph Provider" in the Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal.

Example Activity Graph Provider Scenario

Create a scenario using the following (remember to set the `content-type` of `application/xml`):

Request URI:

```
POST http://localhost:8891/wcps/api/conductor/namespaces/default/scenarios
```

Request Body:

```
<scenarioMetadata>
<scenario xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0">
    <body>
        <call-provider>
            <provider-name>ActivityGraphProvider</provider-name>
            <connection-name>ActivityGraphConfigConnection</connection-name>
            <resource-name>QueryCommonItems</resource-name>
            <method-name>queryCommonItems</method-name>
            <variable>providerResults</variable>
            <input>
                <parameter>
                    <common:name>sourceClassURN</common:name>
                    <common:value>WC.user</common:value>
                </parameter>
                <parameter>
                    <common:name>sourceObjectURN</common:name>
                    <common:value>carl</common:value>
                </parameter>
                <parameter>
                    <common:name>targetClassURN</common:name>
                    <common:value>WC.user</common:value>
```

```
                </parameter>
                <parameter>
                    <common:name>targetObjectURN</common:name>
                    <common:value>monty</common:value>
                </parameter>
                <parameter>
                    <common:name>similarityURN</common:name>
                    <common:value>user-edit</common:value>
                </parameter>
            </input>
        </call-provider>
        <return>
            <expression>${providerResults}</expression>
        </return>
    </body>
    <name>QueryCommonItemsScenario</name>
    <comments>QueryCommonItemsScenario comments</comments>
    <tags>
        <tag>invoke</tag>
        <tag>sample</tag>
        <tag>provider</tag>
        <tag>resource</tag>
        <tag>input</tag>
        <tag>parameters</tag>
        <tag>overloaded</tag>
    </tags>
</scenario>
</scenarioMetadata>
```

Executing the Scenario

Before continuing, you must set the header values: `Accept=application/xml` and `content-type=application/xml`.

Request URI:

```
POST
http://localhost:8891/wcps/api/conductor/namespaces/default/scenarios/QueryCommonI
temsScenario
```

Request Body:

```
<parameters/>
```

Using the AGFunctionProvider

A function provider is a utility class that can be invoked from within a scenario. The `AGFunctionProvider` supports these methods:

```
/**
     * Return the value of 'ObjectURN', only for WC.document, from the input
Object.
     * @param agResults From call to 'QueryRecommendations' or QueryItems.  Can be
one of the following Classes:
     *
     * From AG REST 'recommendations'
     * <ul>
     *   <li> Recommendations
     *   <li> RecommendedItems
     *   <li> List<Recommendation>
     * </ul>
     *
```

```
     * From AG REST 'items'   (common items)
     * <ul>
     *   <li> Results
     *   <li> Items
     *   <li> List<Item>
     * </ul>
     *
     * @param dDocNameOnly  If true, just the last part of the ObjectURN for
     * document will be returned; otherwise, the entire ObjectURN will be.
     * For example, if ObjectURN is example-ucm11g#dDocName:EXAMPLEACL000116,
     * if this value is 'true', then only EXAMPLEACL000116 will be returned.
     * Otherwise, all of stanl18-ucm11g#dDocName:EXAMPLEACL000116 is returned.
     *
     * @return a List of ObjectURN representing the document ID
     */
    @PublicFunction

    public static List<String> getDocumentIDs(
@PublicParameter(parameterName="agResults",descriptionBundleKey="getContentIDs.par
ameter.description.inputList")Object agResults,
 @PublicParameter(parameterName="dDocNameOnly",
descriptionBundleKey="getContentIDs.parameter.description.inputList")boolean
dDocNameOnly)
    {
...

/**
     * Return the value of 'ObjectURN' for the 'ClassURN' from the input Object.
     * @param agResults From call to 'QueryRecommendations' or QueryItems.  Can be
one of the following Classes:
     *
     * From AG REST 'recommendations'
     * <ul>
     *   <li> Recommendations
     *   <li> RecommendedItems
     *   <li> List<Recommendation>
     * </ul>
     *
     * From AG REST 'items'   (common items)
     * <ul>
     *   <li> Results
     *   <li> Items
     *   <li> List<Item>
     * </ul>
     *
     * param filterClassURN return content only of this type.  Examples are:
WC.wiki-page, WC.group-space, WC.user, WC.blog,
     * WC.topic, WC.document.  If this is null, return all types.
     *
     * @return a List of ObjectURN representing the item identifier.
     */
    @PublicFunction
            (
                    functionName="getContentIDs",
                    descriptionBundleKey="getContentIDs.method.description"
            )
    public static List<String>
getContentIDs(@PublicParameter(parameterName="agResults",

descriptionBundleKey="getContentIDs.parameter.description.inputList")Object
```

```
agResults,

@PublicParameter(parameterName="filterClassURN",

descriptionBundleKey="getContentIDs.parameter.description.filterClassURN") String
filterClassURN)
    ...

/**
     * Filter recommendations by score.  Only those recommendations with a score
>= the cutoff will be preserved.
     */
    @PublicFunction
            (
                    functionName="filterRecsByScore",
                    descriptionBundleKey="filterRecsByScore.method.description"
            )
    public static Recommendations filterRecsByScore(
            @PublicParameter(parameterName="recommendations",
            descriptionBundleKey="filterRecsByScore.parameter.description.agRecs")
Recommendations recs,
            @PublicParameter(parameterName="cutoffScore",

descriptionBundleKey="filterRecsByScore.parameter.description.cutoffScore") float
cutoffScore
            )
    {
```

Using the AGFunctionProvider in a Scenario

In the following scenario, a call is first made to the Activity Graph REST service. The results of that call are in the `agRecommendations` variable. The next call in the scenario is to the `AGFunctionProvider.getContentIDs()` method, passing in the results of the `agRecommendations` (an Activity Graph object), and returning a `List<String>` of document IDs, held in the evaluation of the `expressions` call.

```
<expression>${agfunction:getContentIDs(agRecommendations,'WC.user')}</expression>

 <scenario:scenario
xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"

xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
    <comments>CallProviderScenario comments</comments>
    <body>
        <call-provider>
            <provider-name>ActivityGraphProvider</provider-name>
            <connection-name>AgTestServerConnection</connection-name>
            <resource-name>QueryRecommendations</resource-name>
            <method-name>getRecommendationsUsingDefaultRecipe</method-name>
            <variable>agRecommendations</variable>
            <input>
                <parameter>
                    <common:name>classURN</common:name>
                    <common:value>WC.user</common:value>
                </parameter>
                <parameter>
                    <common:name>objectURN</common:name>
                    <common:value>carl</common:value>
                </parameter>
            </input>
        </call-provider>
```

```
          <!-- Now call function provider to translate into user IDs.  A more
specific call is available for WC.document; this just tests
         the ability of the function provider to handle different types of content
-->
        <return>

<expression>${agfunction:getContentIDs(agRecommendations,'WC.user')}</expression>
        </return>
    </body>
    <name>CallProviderScenario</name>
</scenario:scenario>
```

67.2.3 Creating File-Based Scenarios

As well as using the Scenario Editor, you can also create file-based scenarios in XML format. Note that file-based scenarios are only supported through the Conductor's import/export function and JDeveloper MDS integration. To import a file-based scenario into the Scenario Editor, select the **Create from file** option when you create the scenario.

Table 67–1 shows the statements that make up the contents of a scenario file. For more detailed information about the statements, refer to the node descriptions in Section 67.2.2.4, "Specifying Scenario Flow Using Node Types."

Table 67–1 Scenario Statements

Statement	Description	Example

Variable Assignment	Creates a variable that is scoped within the context of the currently running scenario.	`<assign-variable>`
		`<variable>index</variable>`
		`<expression>3</expression>`
		`</assign-variable>`

Execute	Invokes a specified EL Expression with no expected return value. Similar to Variable Assignment, only results of the expression are not stored within the Scenario Context.	`<execute>`
		`<comments>Adds to an existing collection.</comments>`
		
		`<expression>${collections:append(responses,'The outlook is not so good.')}</expression>`
		`</execute>`

Table 67–1 (Cont.) Scenario Statements

Statement	Description	Example

Condition/Otherwise	Evaluates an EL expression and executes the contained statements should the EL expression evaluate to true. Statements within the otherwise block will only execute if all condition statements evaluate to false.	`<conditions>`
		`<body>`
		`<condition>`
		`<body>`
		`<assign-variable>`
		
		`<variable>customerType</variable>`
		`<expression>Great Customer</expression>`
		`</assign-variable>`
		`</body>`
		
		`<expression>${customer.totalMoneySpent >= 100000}</expression>`
		`</condition>`
		`<otherwise>`
		`<body>`
		`<assign-variable>`
		
		`<variable>customerType</variable>`
		`<expression>Good Customer</expression>`
		`</assign-variable>`
		`</body>`
		`</otherwise>`
		`</body>`
		`</conditions>`

ForEach	Provides looping/iterating functionality over a collection of things.	`<for-each>`
		`<body>`
		`<execute>`
		`<comments>`
		`Let's send each great customer an email, thanking them for their loyalty.`
		`</comments>`
		`<expression>`
		
		`${emailService:sendMail(customer,'Subject:Thanks for being a great customer')}`
		`</expression>`
		`</execute>`
		`</body>`
		`<variable>customer</variable>`
		
		`<max-iterations>10</max-iterations>`
		`<items>${greatCustomers}</items>`
		`</for-each>`

Table 67–1 (Cont.) Scenario Statements

Statement	Description	Example

While	Provides looping/iterating functionality as long as the specified EL expression evaluates to true.	`<while>`
		`<body>`
		`<assign-variable>`
		`<comments>Sum up customer invoices to arrive at customer total expenditure.</comments>`
		`<variable>customerTotalSpent</variable>`
		`<expression>${customerTotalSpent+order.invoiceTotal}</expression>`
		`</assign-variable>`
		`</body>`
		`<expression>${customer.id = currentCustomerId }</expression>`
		`<max-iterations>-1</max-iterations>`
		`</while>`

Invoke Scenario	Invokes another named scenario within the same namespace. Results are stored within the context of the currently running scenario.	`<call-scenario >`
		`<variable>greeting</variable>`
		`<scenario>randomGreetingGenerator</scenario>`
		`<input>`
		`<parameter>`
		`<ns2:name>greetingsList</ns2:name>`
		`<ns2:value>hello, Bonjour, Buenos días</ns2:value>`
		`</parameter>`
		`</input>`
		`</call-scenario>`

Invoke Provider	Invokes a named implementation of IDataProvider. Results are stored within the context of the currently running scenario	`<call-provider>`
		`<provider-name>CRM_Provider</provider-name>`
		`<connection-name>crm_db</connection-name>`
		`<resource-name>getGreatCustomers</resource-name>`
		`<variable>greatCustomers</variable>`
		`<input>`
		`<parameter>`
		`<ns2:name>minMoneySpent</ns2:name>`
		`<ns2:value>100000</ns2:value>`
		`</parameter>`
		`</input>`
		`</call-provider>`

Table 67–1 (Cont.) Scenario Statements

Statement	Description	Example

Return	Returns the specified results of evaluation of an EL Expression.	`<return>`
		`  <comments>Return the sales rep who sold the most widgets this month.</comments>`
		
		`<expression>${bestSalesRep.id}</expression>`
		`</return>`

Raise Error	Raise and throw an error with the specified error message	`<raise-error>`
		`  <comments>User is unathorized, so let's return a 401.</comments>`
		`  <statusCode>401</statusCode>`
		`  <message>User ${ScenarioContext.scenarioRequest.userPrincipal.name} is unauthorized to execute this scenario.</message>`
		`</raise-error>`

Example 67–2 Example File-based Scenario

```
<scenario:scenario
xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"

xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <comments>sampleScenarioWithIf comments</comments>
  <body>
    <assign-variable>
      <variable>index</variable>
      <expression>3</expression>
    </assign-variable>
    <conditions>
      <body>
        <condition>
          <body>
            <assign-variable>
              <variable>ResultVar</variable>
              <expression>conditionSatisfied</expression>
            </assign-variable>
          </body>
          <expression>${index == 3}</expression>
        </condition>
        <otherwise>
          <body>
            <assign-variable>
              <variable>ResultVar</variable>
              <expression>otherwiseInvoked</expression>
            </assign-variable>
          </body>
        </otherwise>
      </body>
    </conditions>
    <return>
      <expression>${ResultVar}</expression>
    </return>
  </body>
  <name>sampleScenarioWithIf</name>
  <tags>
```

```
    <tag>if</tag>
    <tag>condition</tag>
    <tag>otherwise</tag>
  </tags>
</scenario:scenario>
```

67.2.4 Displaying Targeted Content at Runtime

This section describes how targeted content is displayed using the Content Presenter to the user at runtime for WebCenter Portal: Spaces and WebCenter Portal: Framework applications. It also describes how you can use REST APIs to create your own user interface for presenting content.

This section contains the following subsections:

- Section 67.2.4.1, "Presenting Content with Content Presenter"
- Section 67.2.4.2, "Presenting Content with a Custom User Interface"

67.2.4.1 Presenting Content with Content Presenter

Content Presenter is tightly integrated with the Conductor's dynamic content query generation and enables simple surfacing of dynamic content. For more information about using Content Presenter in your application, see Chapter 29, "Creating Content Presenter Display Templates."

For information about using Expression Language expressions in WebCenter Portal: Spaces to access a Personalization server scenario, see "Personalizing Pages" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

67.2.4.2 Presenting Content with a Custom User Interface

Using REST APIs, you can create your own user interface for presenting content rather than using Content Presenter. For more information about REST APIs, see Chapter 54, "Using Oracle WebCenter Portal REST APIs."

67.3 Tutorial: Creating, Testing and Deploying a Simple Application

This section describes how to create a simple application in JDeveloper containing a scenario that returns a simple string (`"Hello World"`). You'll then test the results of that scenario by adding it to a JSP page, and run the application in the integrated WebLogic server to display the results in a browser.

This section also covers the steps to configure and start the integrated WebLogic server, define a connection to the Personalization Server Conductor, and how to deploy the scenario and JSP artifacts to the server domain for testing.

> **Note:** as a prerequisite to to this tutorial, you should already have installed WebCenter Portal's extension for Oracle JDeveloper (`oracle.webcenter.framework_bundle.zip`). If you have not already added this extension, use the Update Wizard in JDeveloper to check for updates by clicking **Check for Updates** from the **Help** menu.

1. Create a simple WebCenter Portal application (`Hello_Applicaton`) using the default options and technologies.
   a. From the File menu select New...

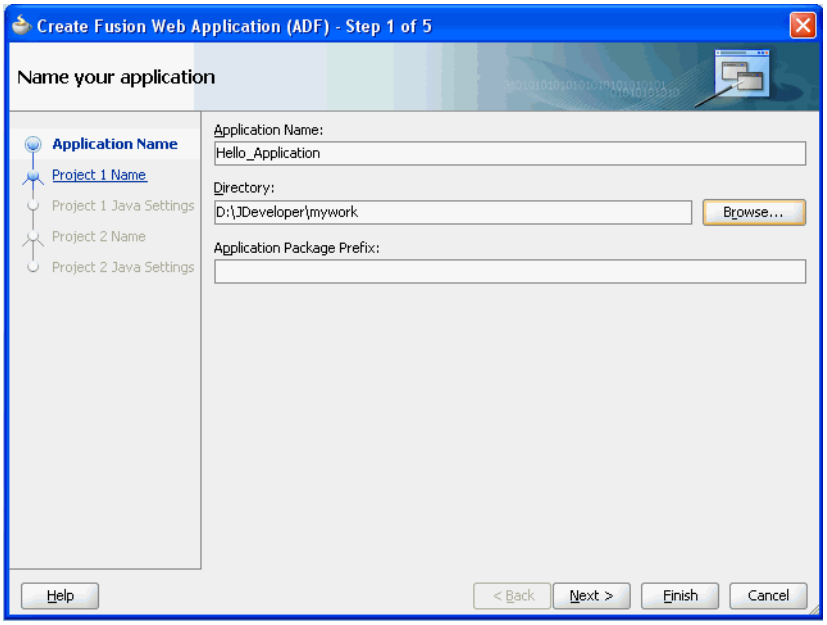
The New Gallery appears.

b. From the All Technologies tab, select Applications.

c. Select `Fusion Web Application (ADF)` from the list and click OK.

The Create Fusion Web Application (ADF) wizard appears (see Figure 67–15).

Figure 67–15 Creating the Hello_Application - Step 1

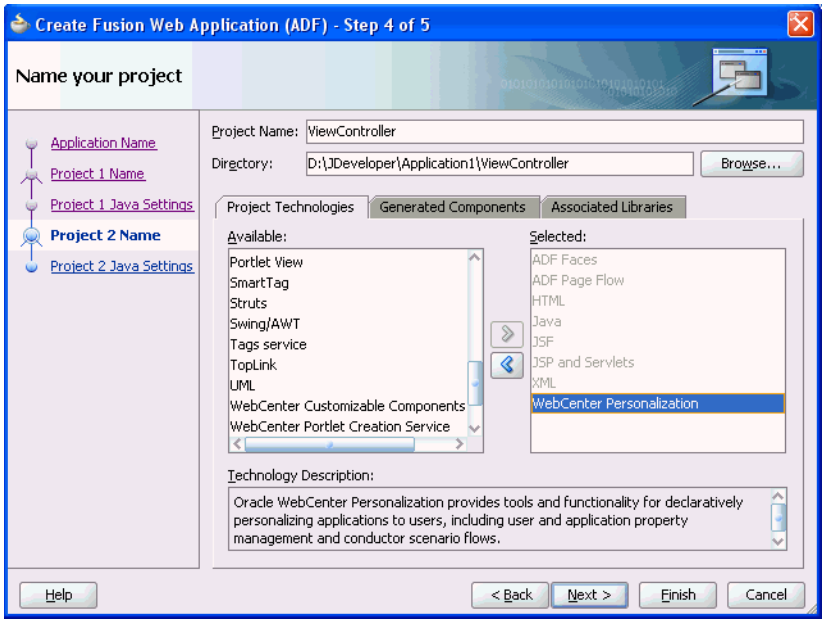


d. Enter the Application Name as `Hello_Application`.

e. Click Next until you arrive at `Step 4 of 5`, the wizard page for the `Portal` project.

The Project Technologies tab displays a list of available technologies to install for the application.

Figure 67–16 Creating the Hello_Application - Step 4



f. Add `Webcenter Personalization` to the list of selected technologies for the `Portal` project, and click Finish to create the configured `Hello_Application` application.

2. Configure JDeveloper's integrated server's (integrated WLS) default domain.

The domain will automatically be extended to include necessary server artifacts required for running the Personalization components (i.e., the Personalization Conductor and the Property Service).

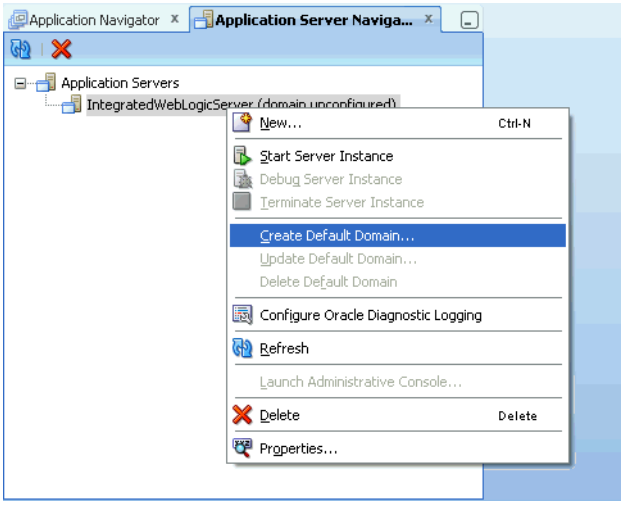
a. From the View menu, select `Application Server Navigator` and expand the `Application Servers` tree node to display the `IntegratedWeblogicServer`.

The current status should be `(domain unconfigured)`.

b. Right-click `IntegratedWeblogicServer` to display the context menu and select `Create Default Domain...`

The status displays `(domain building)` and the Message Log window displays the output as the domain is being built. Wait for the domain to finish building. This may take some time to complete. When complete, the message log should display `"Integrated Weblogic domain processing completed successfully."`

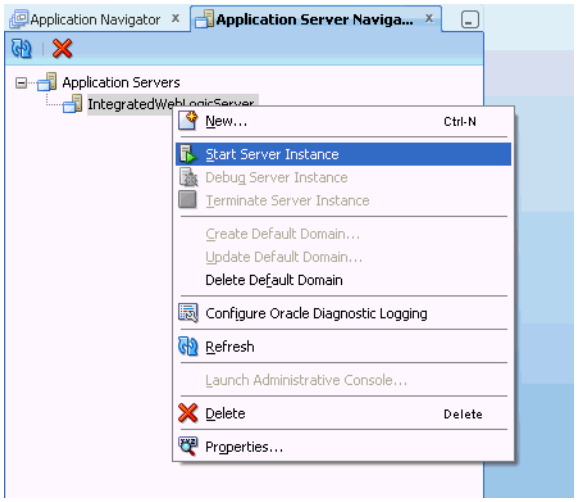
Figure 67–17 Application Server Navigator - Creating the Default Domain



c. Start the server by right-clicking `IntegratedWeblogicServer` to display the context menu, and selecting Start Server Instance.

The `Running: IntegratedWeblogicServer - Log` view should appear and display output as the server is starting. Wait for the server to finish starting. This may take some time to complete. When complete, the IntegratedWeblogicServer Message Log should display `IntegratedWebLogicServer started`.

Figure 67–18 Application Server Navigator - Starting the Server



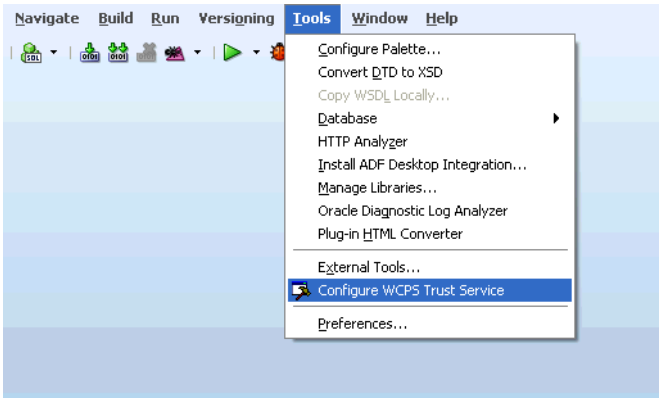
3. Configure the Personalization (WCPS) Trust service on the running server to enable single sign-on (SSO) for JDeveloper's IntegratedWebLogicServer domain.

a. From the Tools menu, select Configure WCPS Trust Service.

The external tool to perform this task will begin running and its output will be directed to the `Configure WCPS Trust Service - Log` view. Wait for the configuration to complete. This may take a while. When complete, the log will display the message `"YOU HAVE MADE CHANGES THAT REQUIRE THE`

```
DOMAIN TO BE RESTARTED. PLEASE RESTART ALL DOMAIN
SERVERS."
```

Figure 67–19 Configuring the WCPS Trust Service



b. Stop the application server by clicking Terminate on the `Running: IntegratedWeblogicServer - Log` view.

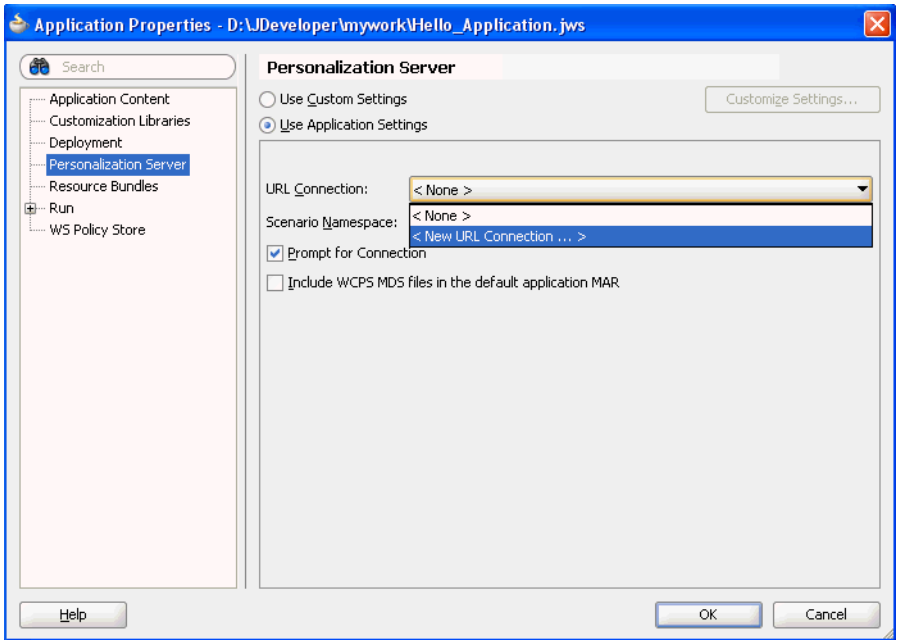
When shutdown is complete you will see the message `"Process Exited"`. You can then proceed to restart the server as shown in step 2c.

4. Configure an application URL connection to the Personalization Conductor component of the application server.

a. From the Application menu select `Application Properties...`

The Application Properties dialog displays.

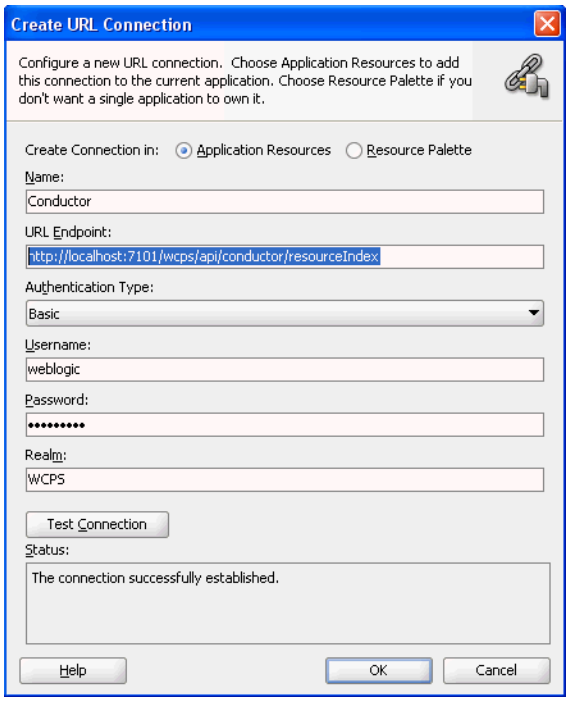
Figure 67–20 Application Properties Dialog - Configuring the Personalization Server Connection



b. Select the `Personalization Server` category, and from the URL Connection: drop-down list and select `< New URL Connection... >`.

The Create URL Connection dialog displays (see Figure 67–21).

Figure 67–21 Create URL Connection Dialog - Conductor URL Connection



c. Enter the (default) data entry fields as shown below to create the connection.

Field Name	Value
Name	Conductor
URL Endpoint	http://localhost:7101/wcps/api/conductor/resourceIndex
Authentication Type	Basic
Username	weblogic
Password	weblogic1
Realm	WCPS

d. Click Test Connection.

The Status should display as `"The connections successfully established."`

e. Click OK to create the Conductor URL connection.

f. On the Personalization Server page, leave the default Scenario Namespace set to the application name (`Hello_Application`), and click OK in the Application Properties dialog to accept the settings.

5. Configure the server deployment descriptor for the `Hello_Application` such that the Personalization Services (WCPS) libraries are available to the application.

a. In the Application Navigator, expand the folder `Portal/Web Content/WEB-INF`.

b. Double-click the `weblogic.xml` deployment descriptor to open it.

   The WebLogic Descriptor Editor displays.

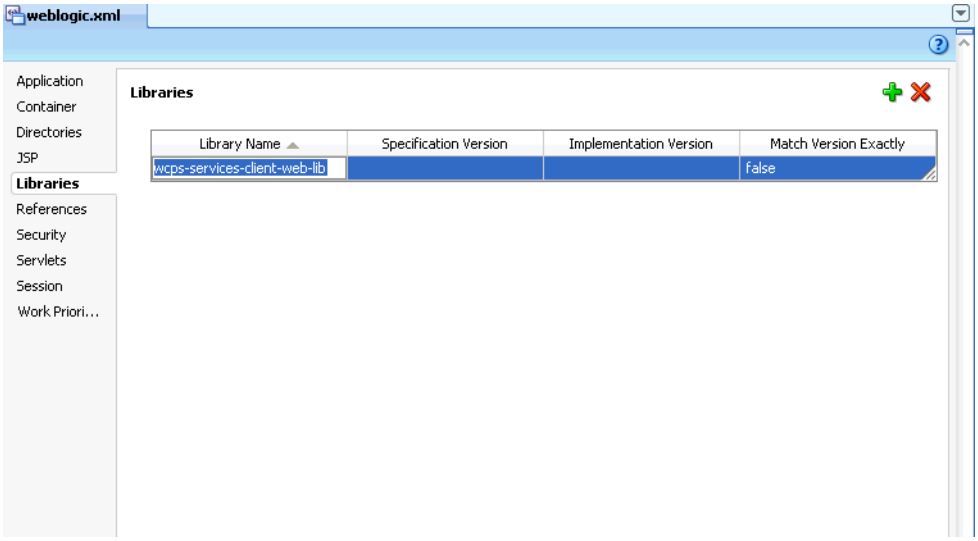
c. Select the `Libraries` category.

d. Click the Plus (+) icon to create a new library entry.

e. In the Library Name column of the table enter the following library name:

   `wcps-services-client-web-lib`

Figure 67–22 WebLogic Descriptor Editor - Adding the WCPS Library



f. When finished, select `Save` from the File menu to save the `weblogic.xml` descriptor.

6. Create a simple `hello_world` scenario that returns the string `"Hello World"` when invoked by the Personalization Conductor.

   a. Right-click the `Portal` project in the Application Navigator and select New...

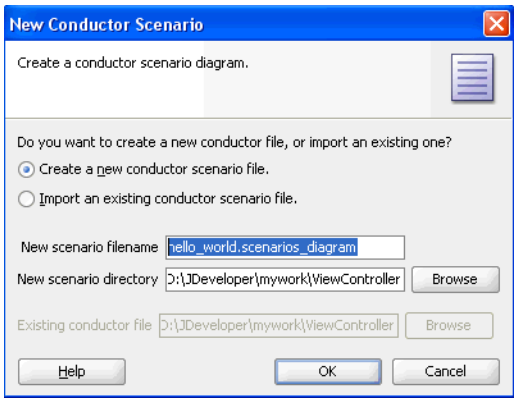
      The New Gallery appears.

   b. From the Current Project Technologies tab, select the `Personalization` category.

   c. Select `Conductor Scenario` and click OK.

      The New Conductor Scenario dialog displays.

Figure 67–23 New Conductor Scenario Dialog



d. Leave the option `Create a new conductor scenario file` selected, enter a New scenario filename called `hello_world.scenarios_diagram`, and click OK to create the scenario.

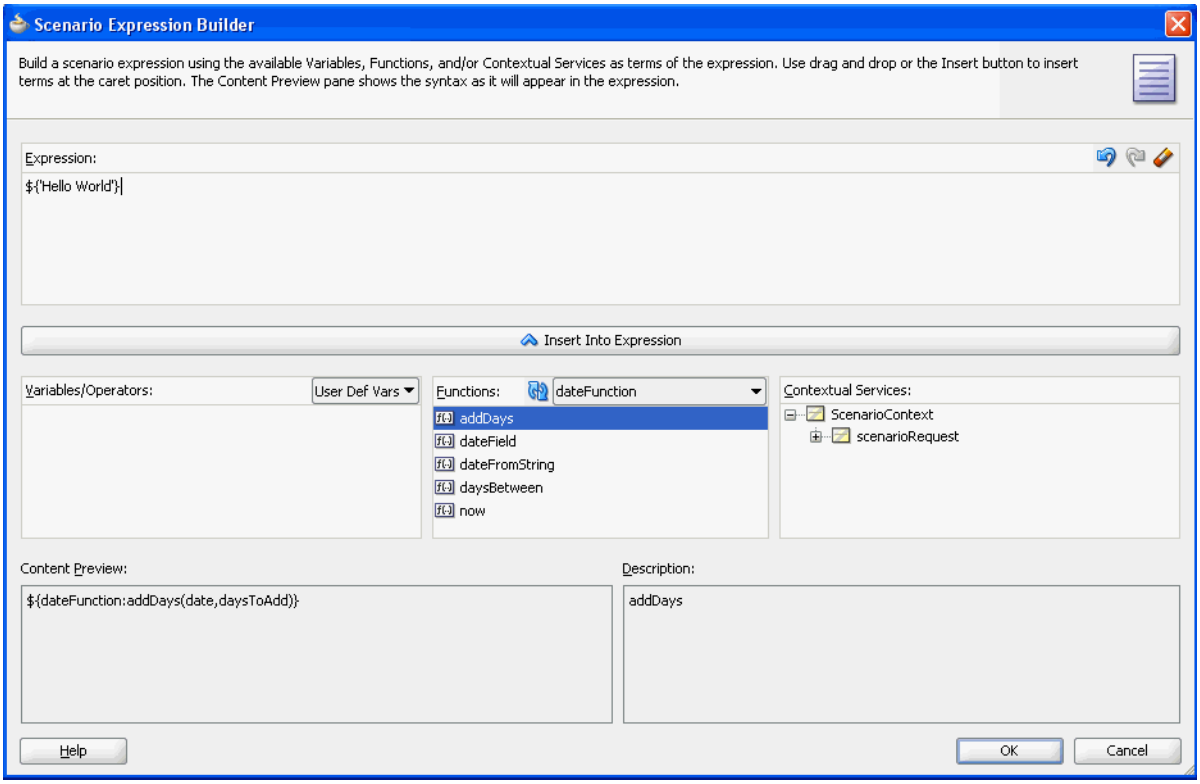
e. In the Scenario Editor's Diagram view, right-click the Start node and select `Add Following Statement->Return`.

A Return node is added and connected.

f. Right-click the Return node and select `Set Expression...`

The Scenario Expression builder displays (see Figure 67–24).

Figure 67–24 Scenario Expression Builder

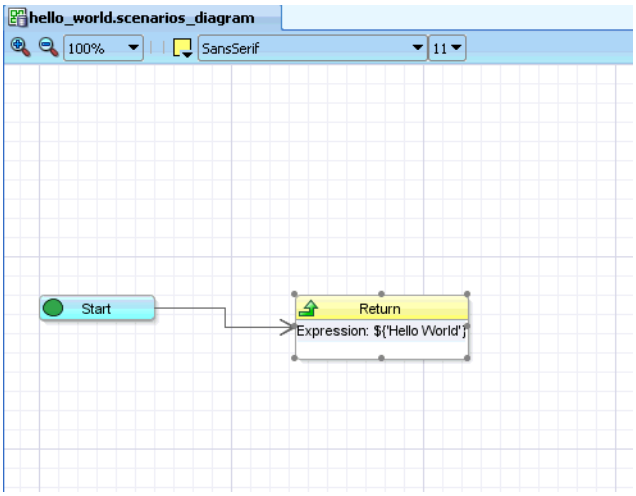


g. In the Expression text area, enter the following EL expression and click OK.

```
${'Hello World'}
```

Your scenario diagram should look like the one shown in Figure 67–25.

Figure 67–25 Scenario Editor - hello_world Scenario



h. From the File menu, select `Save` to save the `hello_world.scenarios_diagram`.

7. Create a JSP page that invokes the `hello_world` scenario and displays the results.

Before creating the JSP page we need to add some libraries to the `Portal` project that contain resources required by the JSP page.

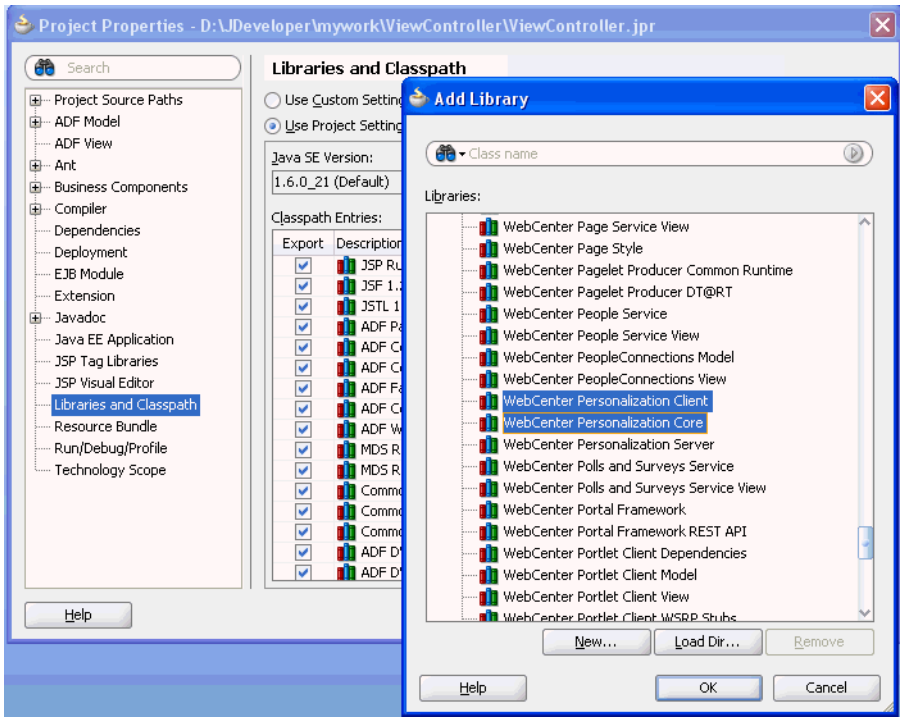
a. Right-click `Portal` project in the Application Naviagtor and select Project Properties...

The Project Properties dialog displays.

b. Select the `Libraries and Classpath` category, and click Add Library...

The Add Library dialog displays (see Figure 67–26).

Figure 67–26 Project Properties Dialog - Adding the Extension Libraries



c. Select the following Extension library from the list and click OK to add it.

```
Webcenter Personalization Client
```

d. Click OK on the Project Properties dialog when finished.

Now we can create the JSP page that executes the `hello_world` scenario on the Personalization Conductor and displays the result.

e. Right-click the `Portal` project in the Application Naviagtor and select New...

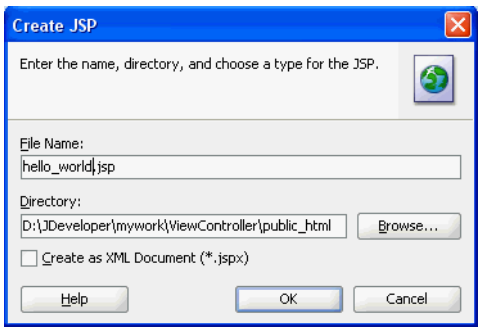
The New Gallery appears.

f. From the Current Project Technologies tab, select the JSP category.

g. From the item list select JSP and click OK.

The Create JSP dialog displays (see Figure 67–27).

Figure 67–27 Create JSP Dialog



h. Enter the File Name as `hello_world.jsp` and click **OK** to create the JSP page.

i. When the JSP Editor launches, select the Source view and replace the template text with the following JSP code:

Notice in the `c:out` command how the Personalization Context Bean `p13nContext` is being used to invoke the Conductor connection (`Conductor`) you created earlier using the scenario namespace `Hello_Application` specified earlier in the Application Properties dialog, invoking the scenario called `hello_world`, and finally displaying the results.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@ page contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
    <title>hello_world</title>
  </head>
  <body>
    <jsp:useBean id="p13nContext"
class="oracle.wcps.client.PersonalizationContext" scope="session"/>

<%((oracle.wcps.client.PersonalizationContext)session.getAttribute("p13nCon
text")).reset(); %>
    <c:out value="${p13nContext.conductor['Conductor'].namespaces['Hello_
application'].scenario['hello_world'].results}"/>
  </body>
</html>
```

Note: If you haven't configured single sign-on and the trust service as outlined in step 3, then the `Conductor` URL connection specified here will generate a 401 "Unauthorized" return code when the page is run. As a shortcut (for testing purposes only) you can substitute an inline connection:

```
http://weblogic:weblogic1@localhost:7101/wcps/api/co
nductor/resourceIndex
```

in place of the `Conductor` URL to get around the single sign-on constraint.

---

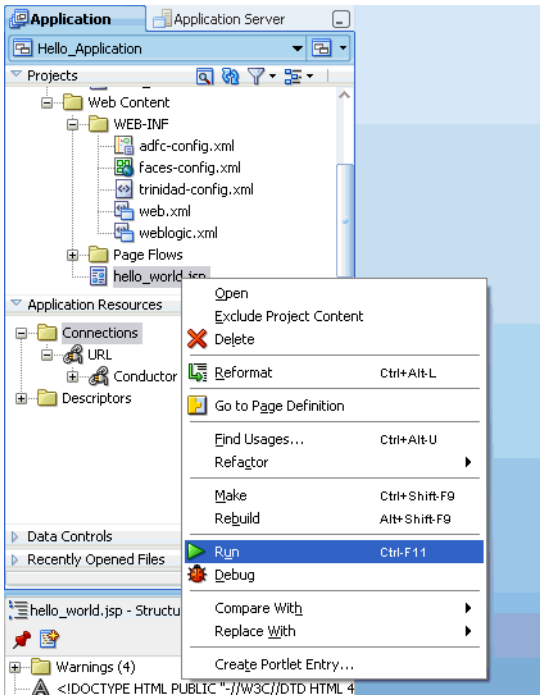
j. From the File menu, select Save to save the `hello_world.jsp`.

8. Finally, to build and deploy the `Hello_Application`, right-click the runnable artifact the `hello_world.jsp` page in the Application Navigator and select Run.

JDeveloper starts several background processes in sequence, including:

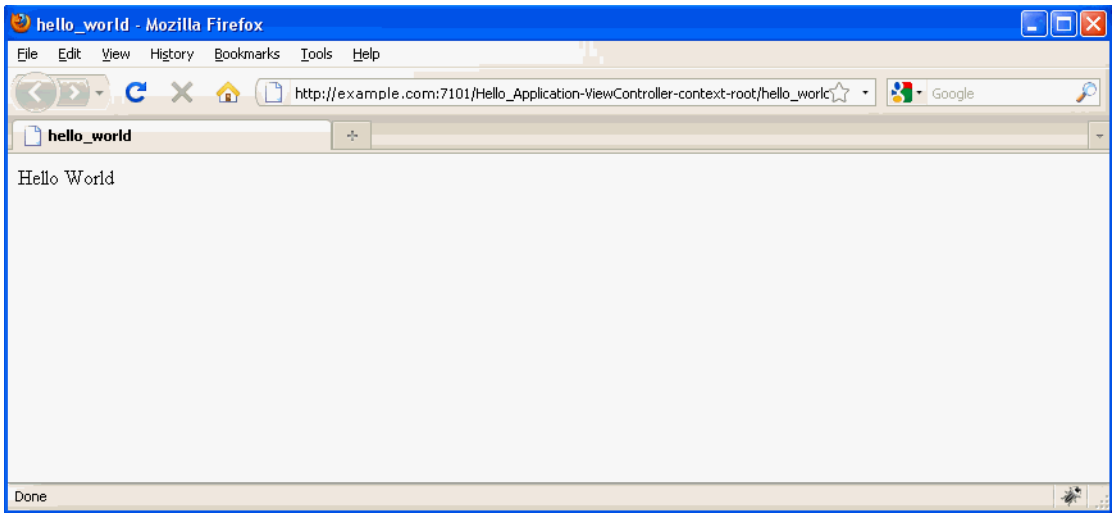
- Building the application,
- Starting the intergated server (if not already started),
- Deploying the application artifacts (including any Personalization artifacts) to the default domain, and
- Launching an external web browser to display the artifact `hello_world.jsp`.

Figure 67–28 Application Navigator - Building and Deploying the Application



The resulting `hello_world.jsp` JSP page appears in the browser.

Figure 67–29 Hello_Application's hello_world.jsp Page Displayed in Browser



67.4 Extending Personalization

This section describes how you can create new ways to personalize application flow or content by adding custom providers and locators.

This section includes the following sub-sections:

- Section 67.4.1, "Adding Custom Providers"

- Section 67.4.2, "Adding Custom Locators"

67.4.1 Adding Custom Providers

Providers provide an extension point to Personalization, with which you can add custom function providers and custom data providers. Table 67–2 shows the available extension points for Function and Data Providers.

Table 67–2 Extension Points

Extension	Description
oracle.wcps.provider.IFunctionProvider	A Function Provider extension point for the Conductor to create utility methods for data manipulation/transformation, business rule calculation, and so forth that can be invoked using EL. A simple example might be: ${strings:concat('string1','string2')}
oracle.wcps.provider.IDataProvider	A Data Provider extension point for the Conductor. Custom data providers can be written to integrate with legacy systems or external data stores.

67.4.1.1 Extending the AnnotatedConfiguration Class

The `AnnotatedConfiguration` class describes the connection parameters required for a data provider, and can be found in:

Location:
`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\connections-service-1.0.jar`

You will need to extend this class and add any other properties your custom provider may require. Create this class first, as the other components will build upon it (see Example 67–3).

Example 67–3 HelloWorldConnectionConfig

```
import oracle.wcps.connection.annotation.ConnectionProperty;
import oracle.wcps.connection.configuration.AnnotatedConfiguration;

@ConnectionConfiguration
(connectionType="hello.world.provider.connection")
@L10n
(bundle="demo.provider.resources.HelloWorldProviderResources")
public class HelloWorldConnectionConfig extends AnnotatedConfiguration
{
  @ConnectionProperty(propertyName="sampleProperty", required=true)
  String sampleProperty;
}
```

67.4.1.2 Implementing the Required Interfaces

IMetadata

Location:
`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\conductor-server-1.0.0.0.jar`

```
public interface IMetadata
{
    public String getName();
    public Set<? extends IMetadata> getMetadata();
}
```

Next, implement the IMetadata interface. This provides information to the tools to surface the connection parameters, essentially representing the information in your `HelloWorldConnectionConfig` class as metadata (see Example 67–4). Each of the properties in the `HelloWorldConnectionConfig` class becomes a child metadata object. This is not so intuitive for simple structures, but is useful when the connection configuration may contain nested classes that relate to other components in the 'model'.

Example 67–4 HelloWorldConfigMetadata

```
public class HelloWorldConfigMetadata implements IMetadata
{
    private String name;
    private Object value ;
    private Map<String, IMetadata> childMDataMap = new HashMap<String,
IMetadata>();

    public HelloWorldConfigMetadata(HelloWorldConnectionConfig config)
    {
        name = config.name;
    }

    public HelloWorldConfigMetadata(String name, Object value)
    {
        this.name = name;
        this.value = value;
    }

    public void addMetadata(IMetadata childMetadata)
    {
        childMDataMap.put(childMetadata.getName(), childMetadata);
    }

    public String getName()
    {
        return name;
    }

    public Object getValue()
    {
        return value;
    }

    public Object getProperty(String propName)
    {
        IMetadata md = childMDataMap.get(propName);
        if (md != null)
        {
            return ((HelloWorldConfigMetadata)md).getValue();
        }
        return null;
    }

    public Set<IMetadata> getMetadata()
    {
        Set<IMetadata> rv = new HashSet<IMetadata>();
        Set<String> childKeySet = childMDataMap.keySet();
        for (String key : childKeySet)
        {
```

```
            rv.add(childMDataMap.get(key));
        }

        return rv;
    }

    public Properties getProperties()
    {
        Properties props = new Properties();
        Set<String> childKeySet = childMDataMap.keySet();
        for (String key : childKeySet)
        {
            HelloWorldConfigMetadata cmd =
                (HelloWorldConfigMetadata)childMDataMap.get(key);
            props.put(key, cmd.getValue());
        }
        return props;
    }

    @Override
    public String toString()
    {
        return (name + ", properties: " + getProperties());
    }
}
```

IExecutableResource

An executable resource is a class having public methods annotated to designate them as executable by the Conductor. These methods could execute code directly, or delegate the call to an external REST service.

Location:

`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\conductor-server-1.0.0.0.jar`

```
public interface IExecutableResource
{
    public Object execute();
}
```

Your `ExecutableResources` can specify additional methods for execution by virtue of this annotation. Note the use of resource bundles (see section on Resource Bundles for more info). You can create any number of `ExecutableResources` and add them to your connection (see Example 67–5).

Example 67–5 SayHelloExecutableResource

```
package demo.provider;

import oracle.wcps.annotation.PublicFunction;
import oracle.wcps.annotation.PublicParameter;
import oracle.wcps.provider.IExecutableResource;

public class SayHelloExecutableResource implements IExecutableResource
{
    // This method will execute if no specific method is given

    public Object execute()
    {
```

```
        return "Hello from my custom provider";
    }

   // This method can be named explicitly in a Scenario to be invoked

    @PublicFunction(functionName = "sayHello",
                    descriptionBundleKey = "provider.method.sayHello.description")
    public String sayHello(@PublicParameter(parameterName = "user",
                                            descriptionBundleKey =

"provider.parameter.sayHello.user.description")
        String user)
    {
        return "Hello, " + user + ".  Enjoy using the new Personalization
feature!";
    }
}
```

IConnection

Once you have implemented your `IExecutableResources` and `IMetadata`, you can construct your actual connection object. Each connection has a specific set of configuration parameters, so your provider could possibly have multiple connection objects (see Example 67–6).

Location:

`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\conductor-server-1.0.0.0.jar`

```
 public interface IConnection
{
    public void setScenarioContext(IScenarioContext context);

    public Map<String, IExecutableResource> getNamedResources();

    public IExecutableResource getDefaultResource();

    public Map<String, IMetadata> getConnectionMetadata();

}
```

Example 67–6 HelloWorldConnection

```
package demo.provider;

import java.util.Collections;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

import java.util.concurrent.ConcurrentHashMap;

import oracle.wcps.provider.IConnection;
import oracle.wcps.provider.IExecutableResource;
import oracle.wcps.provider.IMetadata;
import oracle.wcps.scenario.IScenarioContext;

public class HelloWorldConnection implements IConnection
{
    private IScenarioContext scenarioContext;
```

```
private Map<String, IExecutableResource> executableResources =
    new ConcurrentHashMap<String, IExecutableResource>();

private IExecutableResource defaultResource;
private HelloWorldConfigMetadata configMetadata;
private Map<String, IMetadata> metadataMap = new HashMap<String, IMetadata>();

/**
 * Create metadata about this connection from the incoming config
 * @param config
 */
public HelloWorldConnection(HelloWorldConnectionConfig config)
{
    configMetadata = new HelloWorldConfigMetadata(config);

    // Loop over config properties and add as child metadata
    Iterator<Map.Entry<String, Object>> iter =
        config.entrySet().iterator();
    while (iter.hasNext())
    {
        Map.Entry<String, Object> entry = iter.next();
        String key = entry.getKey();
        Object value = entry.getValue();
        if (value != null)
        {
            configMetadata.addMetadata(new HelloWorldConfigMetadata(key,
                                                                    value));
        }
    }
    metadataMap.put(config.name, configMetadata);

    // Now create the executable resources for this connection
    createExecutableResources(config);
}

public void setScenarioContext(IScenarioContext scenarioContext)
{
    this.scenarioContext = scenarioContext;
}

public Map<String, IExecutableResource> getNamedResources()
{
    return executableResources;
}

public IExecutableResource getDefaultResource()
{
    return defaultResource;
}

public Map<String, IMetadata> getConnectionMetadata()
{
    return metadataMap;
}

/**
 * Create the IExecutableResources for this connection. An executable resource
 * is a class having one or more public methods.
 * @param config
 */
```

```
    private void createExecutableResources(HelloWorldConnectionConfig config)
    {
        // NOTE: you may or may not need to use the config information with your
executable resources

        IExecutableResource helloResource = new SayHelloExecutableResource();
        IExecutableResource addingResource = new AddingExecutableResource();

        executableResources.put("HelloResource", helloResource);
        executableResources.put("AddingResource", addingResource);

        // Make one of them the default
        defaultResource = helloResource;

    }
}
```

An instance of your `HelloWorldConnection` will be instantiated for each entry in your `provider-connections.xml` file that corresponds to the connection type specified in `HelloWorldConnectionConfig` (`connectionType="hello.world.provider.connection"`).

IDataProvider

Finally, we are ready to roll all this into our top-level provider class `HelloWorldProvider`. This is the access point for the Conductor to configure our connections and invoke our `IExecutableResource`(s).

Location:

`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\conductor-server-1.0.0.0.jar`

Project Properties -> Libraries and Classpath -> add Jar/Directory -> (point to above conductor-server-1.0.0.0.jar)

Example 67–7 IDataProvider

```
 public interface IDataProvider
{
    /**
     * Returns a map of connections keyed by connection name
     */
    public Map<String, ? extends IConnection> getNamedConnections();

    /**
     * Returns the "default" connection to use
     */
    public IConnection getDefaultConnection();

    /**
     * Called by the framework to set the current scenario context.
     */
    public void setScenarioContext(IScenarioContext scenarioContext);

    /**
     * Return the annotated class used to house connection configuration.
     *
     * <p>
     * Returning null disables this feature.
     */
    public Class<?> getConnectionConfigClass();
```

```
    /**
     * If getConnectionConfigClass returns a valid configuration class,
     * configurations are provided through this method, if any exist.
     * @param configurations The collection of configurations.
     */
    public void setConnectionConfigurations(List<?> configurations);
}
```

Required annotations:

```
@ContextualProvider
        (
                contextName="HelloWorldProvider",

resourceBundle="demo.provider.resources.HelloWorldProviderResources",
                nameBundleKey="provider.name",
                descriptionBundleKey="provider.description"
        )
```

See Example 67–8 on resource bundles below for more information.

Example 67–8 Resource Bundles

```
package demo.provider;

import java.util.Collections;
import java.util.List;
import java.util.Map;

import java.util.concurrent.ConcurrentHashMap;

import oracle.wcps.annotation.ContextualProvider;
import oracle.wcps.provider.IConnection;
import oracle.wcps.provider.IDataProvider;
import oracle.wcps.scenario.IScenarioContext;
@ContextualProvider
        (
                contextName="HelloWorldProvider",

resourceBundle="demo.provider.resources.HelloWorldProviderResources",
                nameBundleKey="provider.name",
                descriptionBundleKey="provider.description"
        )
public class HelloWorldProvider implements IDataProvider
{
  // Never used, but there is a required public method to set this.
  private IScenarioContext scenarioContext;

  // ConnectionConfigurations may come in via
classes/META-INF/connection-providers.xml
   private List<HelloWorldConnectionConfig> connectionConfigurations;

  // Each IConnection is uniquely identified via the AGConnectionConfig
  private static Map<String, HelloWorldConnection> connections =
              new ConcurrentHashMap<String, HelloWorldConnection>();

  // One of the connections will become the default
  private IConnection defaultConnection;

    public HelloWorldProvider() {
```

```
        super();
    }

    /**
     * Return a map of all our Connections, keyed by config.name.
     * @return
     */
    public Map<String, HelloWorldConnection> getNamedConnections()
    {
        return connections;
    }

    /**
     * One of the connections will be designated the default
     * @return
     */
    public IConnection getDefaultConnection()
    {
        return defaultConnection;
    }

    /**
     * Used to hold parameters specific to a given scenario
     * @param scenarioContext
     */
    public void setScenarioContext(IScenarioContext scenarioContext)
    {
        this.scenarioContext = scenarioContext;
    }

    /**
     * The config class is specific to each data provider
     * @return
     */
    public Class<HelloWorldConnectionConfig> getConnectionConfigClass()
    {
        return HelloWorldConnectionConfig.class;
    }

    /**
     * Create a HelloWorldConnection instance from each of the incoming configs.
     * These configs are created when the provider-connections.xml file is loaded.
     * @param configs
     */
    public void setConnectionConfigurations(List<?> configurations)
    {
        List<HelloWorldConnectionConfig> configs =
(List<HelloWorldConnectionConfig>)configurations;
        for (HelloWorldConnectionConfig config : configs)
        {
            // Create a HelloWorldConnection from the config
            HelloWorldConnection connection = new HelloWorldConnection(config);
            if (config.isDefault)
            {
                defaultConnection = connection;
            }
            connections.put(config.name, connection);
        }
    }
}
```

67.4.1.3 Creating the Resource Files

These files also go in the same directory as your classes, so they will be added to your final JAR file. The location of the files must correspond to the package name given in `HelloWorldProvider:bundle="demo.provider.resources.HelloWorldProviderResources"`

```
File name: HelloWorldProviderResources.properties

File contents: provider.name=Hello World
provider.description=Data provider for Hello World

provider.parameter.sayHello.user.description=Method to say hello to a named user
provider.parameter.addTwoIntegers.int1.description=First number to add
provider.parameter.addTwoIntegers.int2.description=Second number to add
```

67.4.1.4 Wiring the Data Provider to the Conductor

Configure your connection in as described in

Name your provider for the Conductor by creating this file in the `META-INF/services` directory (in the same directory structure as your classes, so they will be added to your final jar file):

```
file name: oracle.wcps.provider.IDataProvider
file contents: demo.provider.HelloWorldProvider (fully-qualified name of your
provider)
```

67.4.1.5 Invoking your Data Provider in a Scenario

Note the resource name is the same as in your `HelloWorldConnections` source code.

Example 67–9 Scenario Calling the Provider

```
<scenario:scenario
xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"

xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
    <body>
        <call-provider>
            <provider-name>HelloWorldProvider</provider-name>
            <connection-name>HelloWorldConnection</connection-name>
            <resource-name>HelloResource</resource-name>
            <method-name>sayHello</method-name>
            <variable>providerResults</variable>
            <input>
                <parameter>
                    <common:name>user</common:name>
                    <common:value>cindymc</common:value>
                </parameter>
            </input>
        </call-provider>
        <return>
            <expression>${providerResults}</expression>
        </return>
    </body>
    <name>TestMyProviderScenario</name>
</scenario:scenario>
```

67.4.2 Adding Custom Locators

This section describes how you can extend your personalized WebCenter Portal application by adding custom locators.

This section contains the following subsections:

- Section 67.4.2.1, "Understanding Property Locators"
- Section 67.4.2.2, "Implementing the Required Interfaces"
- Section 67.4.2.3, "Implementing Security"
- Section 67.4.2.4, "Creating a Property Set Definition"
- Section 67.4.2.5, "Putting It All Together"
- Section 67.4.2.6, "Invoking the Property Locator"

67.4.2.1 Understanding Property Locators

A property locator is invoked by the Property Service to access external stores of properties. For example, Personalization provides an out-of-the-box `PeoplePropertyLocator` that uses the People Connections REST service to access profile information for users. The locator is not invoked directly by the Conductor/Scenario, but is instantiated by the Property Set service, which then delegates calls for retrieving properties and property sets to the underlying locator implementation.

The example in the following sections will walk you through creating your own `BookPropertyLocator` to find a book by title in a library, and load its attributes into a property set you have created to represent a Book object.

There are two major steps to implementing this example:

1. Define the PropertySet for Book, and let the Property Set service know about it
2. Implement the `IPropertyLocator` to load properties from a Book object.

67.4.2.2 Implementing the Required Interfaces

Location:

`\Oracle\Middleware\jdeveloper\webcenter\modules\wcps-services_11.1.1.4.0\properties-common-1.0.0.0.jar`

```
public interface IPropertyLocator
{
    public void loadProperties(IContext context, IPropertySet propertySet,
                               IPropertySetDefinition propertySetDefinition,
                               List<IPropertyName> propertyNames);

    public void storeProperties(IContext context,  propertySet,
                                IPropertySetDefinition propertySetDefinition,
                                List<IPropertyName> propertyNames);

    public void removeProperties(IContext context,  propertySet,
List<IPropertyName> propertyNames);

    public void removeProperties(IContext context,  nameSpaceName,
IPropertySetDefinitionName name);

    public IPagedList<IPropertySet> filter(IContext context,  nameSpaceName,
                                           IPropertySetDefinitionName
setDefinitionName, PropertySetFilterContext filterContext);
```

```
    public int count(IContext context,  nameSpaceName, PropertySetExpression
expression);
}
```

67.4.2.3 Implementing Security

The `IContext` object is the vehicle for authentication, passed in from the Property Service. There are two ways you can access the credentials:

- Using the request: `HttpServletRequest request = context.getProperty(IContext.HTTP_REQUEST_KEY)`
- Using the user/password: `String user = context.getProperty(IContext.USERNAME_KEY); String pw = context.getProperty(IContext.PASSWORD_KEY);`

67.4.2.4 Creating a Property Set Definition

This section describes how to create a `PropertySetDefinition` that will be used with your locator.

Follow the steps below to create the `PropertySetDefinition`:

1. Create a new JDeveloper project for your source files.
2. Add the required Personalization classes to the project classpath.
   a. Right-click on the project name and select **Project Properties**.
   b. Select **Libraries and Classpath**, then select **Add Library**.
   c. Enter the name of the library to search for (for example, IPropertyLocator) and select it when it appears.
   d. Repeat these steps for other classes your source code will need to reference.
3. Compile the project and deploy it to a JAR file.
4. Copy the JAR file to the Personalization extensions directory in `/DefaultDomain/locator-extensions-library/WEB-INF/lib`.

67.4.2.5 Putting It All Together

This section describes how to implement the property locator, data object, and data layer.

Implement the IPropertyLocator

(read-only for demo simplicity)

```
/**
 * Our locator will be read-only
 */
public class BookPropertyLocator implements IPropertyLocator
{
    // Let's delegate all the hard work to a BookPropertyDAO
    private BookPropertyDAO dao;

    // Must have a public default constructor, since the Property Service
instantiates
    // this

    public BookPropertyLocator()
    {
```

```
        // todo: show how connection properties could be initialized and passed in
        dao = new BookPropertyDAO();
    }

    /**
     * Load property values, optionally returning only a subset of those.
     * @param iPropertySet The PropertySet into which the retrieved values will be
loaded
     * @param iPropertySetDefinition The PropertySetDefinition defining the
'shape' of the properties
     * @param list List of PropertyNames to load.  If null, all will be loaded.
     */
    public void loadProperties(IContext context, IPropertySet iPropertySet,
                               IPropertySetDefinition iPropertySetDefinition,
                               List<IPropertyName> list)
    {
        try
        {
            // Delegate to BookProperDAO
            dao.loadProperties(iPropertySet, list);
        } catch (Exception e)
        {
            throw new RuntimeException("Cannot load properties ", e);
        }
    }


    /**
     * Filter properties; load/return those properties that only match the filter
criteria.
     * @param iNameSpaceName
     * @param iPropertySetDefinitionName
     * @param propertySetFilterContext
     * @return
     */
    public IPagedList<IPropertySet> filter(IContext context, INameSpaceName
iNameSpaceName,
                                           IPropertySetDefinitionName
iPropertySetDefinitionName,
                                           PropertySetFilterContext
propertySetFilterContext)
    {
        return null;   // todo: possibly implement for demo
    }

    public int count(IContext context, INameSpaceName iNameSpaceName,
                     PropertySetExpression propertySetExpression)
    {
        return 0;
    }


    // As a read-only implementation, we don't support this
    public void storeProperties(IContext context, IPropertySet iPropertySet,
                                IPropertySetDefinition iPropertySetDefinition,
                                List<IPropertyName> list)
    {
        throw new UnsupportedOperationException("BookPropertyLocator is
read-only");
    }
```

```
    // As a read-only implementation, we don't support this
    public void removeProperties(IContext context, IPropertySet iPropertySet,
                                 List<IPropertyName> list)
    {
        throw new UnsupportedOperationException("BookPropertyLocator is
read-only");
    }

    // As a read-only implementation, we don't support this
    public void removeProperties(IContext context, INameSpaceName iNameSpaceName,
                                 IPropertySetDefinitionName
iPropertySetDefinitionName)
    {
        throw new UnsupportedOperationException("BookPropertyLocator is
read-only");
    }
}
```

Implement the Data Access Object for retrieving property values

```
/**
 * Data access class, to load books from a backing store (perhaps a database or
 * some REST service)
 */
class BookPropertyDAO
{
    // Generate the library that will provide our backing store.
    static
    {
        Library.generateLibrary();
    }

    /**
     * Assume the name of the PropertySet is the title of the book.
     * @param propertySet
     * @param list
     */
    void loadProperties(IContext context, IPropertySet propertySet,
                        List<IPropertyName> list) throws Exception
    {
        String title = propertySet.getPropertySetName().getName();
        Book book = Library.getBook(title);
        loadProperties(book, propertySet, list);
    }

    private void loadProperties(IContext context, Book book, IPropertySet
propertySet,
                                List<IPropertyName> propertyNames) throws
Exception
    {
        // Loop over all property names, create a Property value, and add it to
the Property set
        for (IPropertyName pname : propertyNames)
        {
            // System.err.println("Getting property: " + pname + ", with name: " +
pname.getName());
            String value = book.getProperty(pname.getName());
            if (value != null)
            {
                addProperty(value, pname, propertySet);
```

```
            }
        }
    }

    private void addProperty(IContext context, String value, IPropertyName pname,
                             IPropertySet propertySet)
    {

        IProperty<String> ps =
            Property.<String>builder().withProperty(pname,
value).buildWithOutValidation();
        propertySet.putProperty(pname, ps);
    }
}
```

Implement the data layer (Books within a Library)

```
 /**
 * Simple class to demonstrate loading properties.  It could be improved by adding
 * support for non-String attributes (Date, arrays, etc) and using Java Bean
 * Introspection/Reflection to get the property values.
 */
public class Book
{
    String title;
    String author;
    String ISBN;

    public Book(String title, String author, String ISBN)
    {
      setTitle(title);
      setAuthor(author);
      setISBN(ISBN);
    }

    public void setTitle(String title)
    {
        this.title = title;
    }

    public String getTitle()
    {
        return title;
    }

    public void setAuthor(String author)
    {
        this.author = author;
    }

    public String getAuthor()
    {
        return author;
    }

    public void setISBN(String ISBN)
    {
        this.ISBN = ISBN;
    }

    public String getISBN()
```

```
    {
        return ISBN;
    }

    String getProperty(String propName)
    {
         // Here, we could use introspection on the Book class, but since there
are only
         // three properties, we can code it simpler
         if (propName.equals("ISBN"))
         {
           return getISBN();
         }
         if (propName.equals("title"))
         {
           return getTitle();
         }
         if (propName.equals("author"))
         {
           return getAuthor();
         }
         return null;  // todo: perhaps throw Exception here
    }
}

/**
 * Class to generate a library of Books
 */
class Library
{
    static String [] titles = {"Into Thin Air", "Climbing Everest", "Rocky
Mountains"};
    static String [] ISBNs = {"0-201-69581-2", "0-423-78123-1", "0-321-23425-2"};
    static String [] authors = {"Jon Krakauer", "Fooey Handley", "Rick Bicknell"};

    // Keyed by title
    private static Map<String, Book> library = new HashMap<String,Book>();

    static void generateLibrary()
    {
        for (int i=0; i<titles.length; i++)
        {
            Book book = new Book(titles[i], authors[i], ISBNs[i]);
            library.put(book.getTitle(), book);
        }
    }

   static Book getBook(String title)
   {
     return library.get(title);
   }
}
```

67.4.2.6 Invoking the Property Locator

The `PropertyLocator` is invoked by the Property Service, which is itself part of the `PropertiesServiceProvider`. We can invoke the `GetProperties` method on that provider in a scenario, as shown below:

```
<scenario:scenario
xmlns:common="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
```

```
xmlns:scenario="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <body>
  <call-provider>
      <provider-name>oracle.PropertiesServiceProvider</provider-name>
      <method-name>GetProperty</method-name>
      <variable>property</variable>
      <input>
        <parameter>
            <common:name>namespace</common:name>
            <common:value>book.property.locator</common:value>
        </parameter>
        <parameter>
            <common:name>definition</common:name>
            <common:value>BookPropertySetDefinition</common:value>
        </parameter>
        <parameter>
            <common:name>set</common:name>
            <common:value>Into Thin Air</common:value>
        </parameter>
        <parameter>
            <common:name>property</common:name>
            <common:value>ISBN</common:value>
        </parameter>
      </input>
    </call-provider>
    <return>
      <expression>${property}</expression>
    </return>
  </body>
  <name>BookPropertyScenario</name>
</scenario:scenario>
```

68

Using Personalization APIs Externally

A client can access the WebCenter Personalization public APIs using Java APIs or public Rest APIs as described in the following sections:

- Section 68.1, "Using Java APIs"
- Section 68.2, "Using the Property Service REST APIs"
- Section 68.3, "Property Service REST APIs"
- Section 68.4, "Using the Conductor REST APIs"
- Section 68.5, "Calling Personalization Client Services Using ELs"

68.1 Using Java APIs

The service interfaces shown in Table 68–1 define the remote Conductor API.

Table 68–1 Remote Conductor APIs

Interface/Service	Description
oracle.wcps.conductor.services.IScenariosService	General management, retrieval, and execution of stored Scenarios.
oracle.wcps.conductor.services.INamespaceService	General management of stored namespaces known to the Conductor.
oracle.wcps.conductor.services.IProviderService	Retrieval of provider metadata known to the Conductor server.

For more information about Personalization Java APIs, see the JavaDoc for WebCenter in the *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

68.2 Using the Property Service REST APIs

This section provides examples of how you can use the Property Service's REST APIs to programatically carry out basic Personalization tasks, such as:

- Creating a namespace
- Viewing and creating property set definitions
- Viewing and creating property definitions
- Viewing, creating, and executing scenarios
- Viewing data provider details

This section contains the following subsections:

- Section 68.2.1, "Accessing the Property Service's Resource Index"
- Section 68.2.2, "Creating a Namespace Using the Property Service"
- Section 68.2.3, "Viewing Property Set Definitions Within a Namespace Using the Property Service"
- Section 68.2.4, "Creating a Property Definition Within a Namespace Using the Property Service"
- Section 68.2.5, "Viewing Property Definitions Within a Namespace Using the Property Service"
- Section 68.2.6, "Creating a Property Set Definition Within a Namespace Using the Property Service"
- Section 68.2.7, "Creating a Property Set Within a Namespace Using the Property Service"
- Section 68.2.8, "Viewing a Property Set Within a Namespace Using the Property Service"
- Section 68.2.9, "Accessing the Conductor ResourceIndex"
- Section 68.2.10, "Accessing Namespaces Using the Conductor"
- Section 68.2.11, "Creating a Scenario Within a Namespace Using the Conductor"
- Section 68.2.12, "Viewing Scenarios Within a Namespace Using the Conductor"
- Section 68.2.13, "Executing a Scenario Using the Conductor"
- Section 68.2.14, "Viewing Data Providers Within a Namespace Using the Conductor"
- Section 68.2.15, "Viewing Detailed Data Provider Properties Within a Namespace Using the Conductor"
- Section 68.2.16, "Viewing Data Provider Connections Within a Namespace Using the Conductor"
- Section 68.2.17, "Viewing Data Provider Connection Resources Within a Namespace Using the Conductor"
- Section 68.2.18, "Viewing Data Provider Connection Resources Method Within a Namespace Using the Conductor"
- Section 68.2.19, "Viewing Function Providers Within a Namespace Using the Conductor"

68.2.1 Accessing the Property Service's Resource Index

The Property Service resource index's response contains links and templates to access other resources of the Property Service. For the request and response for accessing the Property Service's resource index, see Section 68.3.1.1, "ResourceIndex."

68.2.2 Creating a Namespace Using the Property Service

Creating an object using the REST HTTP interface requires setting the request method to `POST` and adding a header `content-type` with the value `application/xml`. For the request and response for creating a namespace using the Property Service, see Section 68.3.2.1, "Create Namespace."

68.2.3 Viewing Property Set Definitions Within a Namespace Using the Property Service

For the request and response for viewing property set definitions within a namespace using the Property Service, see Section 68.3.4.3, "Retrieve All Property Set Definitions."

68.2.4 Creating a Property Definition Within a Namespace Using the Property Service

For the request and response for creating a property definitions within a namespace using the Property Service, see Section 68.3.3.1, "Create Property Definition - Create Integer Property Definition."

68.2.5 Viewing Property Definitions Within a Namespace Using the Property Service

For the request and response for viewing property definitions within a namespace using the Property Service, see Section 68.3.3.6, "Retrieve all Property Definitions."

68.2.6 Creating a Property Set Definition Within a Namespace Using the Property Service

For the request and response for creating a property set definition within a namespace using the Property Service, see Section 68.3.4.1, "Create Property Set Definition."

68.2.7 Creating a Property Set Within a Namespace Using the Property Service

For the request and response for creating a property set within a namespace using the Property Service, see Section 68.3.5.1, "Create Property Set."

68.2.8 Viewing a Property Set Within a Namespace Using the Property Service

For the request and response for viewing a property set within a namespace using the Property Service, see Section 68.3.5.5, "Retrieve all Property Sets."

68.2.9 Accessing the Conductor ResourceIndex

Note that in the example below the link templates and ResourceIndex will be modified to comply with best practice guidelines.

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/resourceIndex
```

Response:

```
<resourceIndex resourceType="urn:oracle:wcps:conductor:resourceIndex">
  <links>
    <link resourceType="urn:oracle:wcps:conductor:namespaces"
rel="urn:oracle:wcps:conductor:namespaces"
href="http://localhost:8891/wcps/api/conductor/namespaces"
capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D"
resourceType="urn:oracle:wcps:conductor:namespace"
rel="urn:oracle:wcps:conductor:namespace"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/scen
arios?tag={tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsP
erPage={itemsPerPage}" resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
```

```
rel="urn:oracle:wcps:conductor:scenarioMetadatas"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/scen
arios/%7Bscenario-name%7D?projection={projection}&amp;startIndex={startIndex}&amp;
itemsPerPage={itemsPerPage}" resourceType="urn:oracle:wcps:conductor:scenario"
rel="urn:oracle:wcps:conductor:scenario"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={it
emsPerPage}" resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:wcps:conductor:provider:dataProviders"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
rel="urn:oracle:wcps:conductor:provider:dataProvider"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/func
tionProviders?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage
={itemsPerPage}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:wcps:conductor:provider:functionProviders"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/func
tionProviders/%7Bcategory-name%7D:%7Bfunction-name%7D?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
rel="urn:oracle:wcps:conductor:provider:functionProvider"
capabilities="urn:oracle:restframework:read"/>
  </links>
</resourceIndex>
```

68.2.10 Accessing Namespaces Using the Conductor

Request URI:

```
GET  http://localhost:8891/wcps/api/conductor/namespaces
```

Response:

```
<namespaces resourceType="urn:oracle:wcps:conductor:namespaces">
  <links>
    <link resourceType="urn:oracle:wcps:conductor:resourceIndex"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/resourceIndex"
capabilities="urn:oracle:restframework:read"/>
    <link resourceType="urn:oracle:wcps:conductor:namespaces" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces"
capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"/>
  </links>
  <items>
    <namespace resourceType="urn:oracle:wcps:conductor:namespace">
      <links>
        <link resourceType="urn:oracle:wcps:conductor:namespaces"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces"
capabilities="urn:oracle:restframework:read"/>
        <link resourceType="urn:oracle:wcps:conductor:namespace" rel="self"
```

```
href="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.perso
n" capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.p
erson/scenarios?tag={tag}&amp;projection={projection}&amp;startIndex={startIndex}&
amp;itemsPerPage={itemsPerPage}"
resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
rel="urn:oracle:wcps:conductor:scenarioMetadatas"
href="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.perso
n/scenarios" capabilities="urn:oracle:restframework:read
urn:oracle:restframework:create"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.p
erson/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:wcps:conductor:provider:dataProviders"
href="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.perso
n/dataProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.p
erson/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:wcps:conductor:provider:functionProviders"
href="http://localhost:8891/wcps/api/conductor/namespaces/people.connections.perso
n/functionProviders" capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>people.connections.person</name>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:conductor:namespace">
      <links>
        <link resourceType="urn:oracle:wcps:conductor:namespaces"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces"
capabilities="urn:oracle:restframework:read"/>
        <link resourceType="urn:oracle:wcps:conductor:namespace" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/JDEV"
capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/scenarios?tag={
tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={ite
msPerPage}" resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
rel="urn:oracle:wcps:conductor:scenarioMetadatas"
href="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/scenarios"
capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/dataProviders?p
rojection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:wcps:conductor:provider:dataProviders"
href="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/dataProviders"
capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/functionProvide
rs?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:wcps:conductor:provider:functionProviders"
href="http://localhost:8891/wcps/api/conductor/namespaces/JDEV/functionProviders"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>JDEV</name>
```

```
    </namespace>
  </items>
</namespaces>
```

68.2.11 Creating a Scenario Within a Namespace Using the Conductor

Request URI:

```
POST http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios
```

Request Body:

```
<scenarioMetadata>
<scenario xmlns:ns2="http://xmlns.oracle.com/webcenter/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/webcenter/conductor/scenarios/1.0.0">
  <body>
    <assign-variable>
      <variable>var1</variable>

<expression>${ScenarioContext.scenarioRequest.parameters['param1']}</expression>
    </assign-variable>
    <assign-variable>
      <variable>var2</variable>
      <expression>${var1}</expression>
    </assign-variable>
    <return>
      <expression>${var2}</expression>
    </return>
  </body>
  <name>FeatureCompleteScenario</name>
  <comments>FeatureCompleteScenario comments</comments>
  <tags>
    <tag>FeatureScenario</tag>
    <tag>ScenarioExample</tag>
    <tag>Simple</tag>
  </tags>
</scenario>
</scenarioMetadata>
```

Response:

```
<scenarioMetadata xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario?projection={projection}&startIndex={st
artIndex}&itemsPerPage={itemsPerPage}"
resourceType="urn:oracle:wcps:conductor:scenario-metadata" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios" capabilities="urn:oracle:restframework:read
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete urn:oracle:restframework:execute"/>
    <link resourceType="urn:oracle:wcps:conductor:scenario" rel="alternate"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario/definition"
capabilities="urn:oracle:restframework:read"/>
  </links>
  <status>PUBLISHED</status>
  <created-author>anonymous</created-author>
  <created-date-time>2010-05-11T15:52:30.729-07:00</created-date-time>
  <updated-author>anonymous</updated-author>
```

```
  <updated-date-time>2010-05-11T15:52:30.729-07:00</updated-date-time>
  <scenario>
    <type>Scenario</type>
    <body>
      <assign-variable>
        <type>VariableAssignment</type>
        <variable>var1</variable>

<expression>${ScenarioContext.scenarioRequest.parameters['param1']}</expression>
      </assign-variable>
      <assign-variable>
        <type>VariableAssignment</type>
        <variable>var2</variable>
        <expression>${var1}</expression>
      </assign-variable>
      <return>
        <type>Return</type>
        <expression>${var2}</expression>
      </return>
    </body>
    <name>FeatureCompleteScenario</name>
    <comments>FeatureCompleteScenario comments</comments>
    <tags>
      <tag>FeatureScenario</tag>
      <tag>ScenarioExample</tag>
      <tag>Simple</tag>
    </tags>
    <input-parameters/>
  </scenario>
</scenarioMetadata>
```

68.2.12 Viewing Scenarios Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios
```

Response:

```
<scenarios xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios?tag={tag}&projection={projection}&startIndex={startIndex}&item
sPerPage={itemsPerPage}"
resourceType="urn:oracle:wcps:conductor:scenarios" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios" capabilities="urn:oracle:restframework:read
urn:oracle:restframework:read urn:oracle:restframework:create"/>
  </links>
  <items>
    <scenarioMetadata>
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario?projection={projection}&startIndex={st
artIndex}&itemsPerPage={itemsPerPage}"
resourceType="urn:oracle:wcps:conductor:scenario-metadata" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario"
```

```
capabilities="urn:oracle:restframework:read urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute"/>
        <link resourceType="urn:oracle:wcps:conductor:scenario" rel="alternate"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario/definition"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <status>PUBLISHED</status>
      <created-author>anonymous</created-author>
      <created-date-time>2010-05-11T15:52:30.729-07:00</created-date-time>
      <updated-author>anonymous</updated-author>
      <updated-date-time>2010-05-11T15:52:30.729-07:00</updated-date-time>
      <scenario>
        <type>Scenario</type>
        <name>FeatureCompleteScenario</name>
        <comments>FeatureCompleteScenario comments</comments>
        <tags>
          <tag>FeatureScenario</tag>
          <tag>ScenarioExample</tag>
          <tag>Simple</tag>
        </tags>
        <input-parameters/>
      </scenario>
    </scenarioMetadata>
  </items>
</scenarios>
```

68.2.13 Executing a Scenario Using the Conductor

Request URI:

```
POST http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/scenarios/FeatureCompleteScenario
```

Request Body:

```
<parameters>
  <parameter>
    <name>param1</name>
    <value>param1Value</value>
</parameter>
<parameter>
    <name>param2</name>
    <value>param2Value</value>
  </parameter>
</parameters>
```

Response:

```
<conductorExecutionResults resourceType="urn:oracle:wcps:conductor:genericType">
  <type>java.lang.String</type>
  <results xsi:type="xs:string"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema">param1Value</results>
</conductorExecutionResults>
```

68.2.14 Viewing Data Providers Within a Namespace Using the Conductor

Three IDataProvider implementations are provided out-of-the-box and are accessible in every namespace: `ActivityGraphProvider`; `CMISDataProvider`; `PropertiesProvider`.

Request URI:

```
http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders
```

Response:

```
<dataProviders resourceType="urn:oracle:wcps:conductor:provider:dataProviders">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders" capabilities="urn:oracle:restframework:read"/>
  </links>
  <items>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/CMISProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/CMISProvider"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>CMISProvider</name>
      <localizedName>Universal Content Provider (CMIS)</localizedName>
      <localizedDescription>Provides access to content over CMIS
connections.</localizedDescription>
    </dataProvider>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/ActivityGraphProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/ActivityGraphProvider"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>ActivityGraphProvider</name>
      <localizedName>Activity Graph</localizedName>
      <localizedDescription>Activity Graph Adapter</localizedDescription>
    </dataProvider>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
```

```
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>oracle.PropertiesServiceProvider</name>
      <localizedName>Property Service Data Provider</localizedName>
      <localizedDescription>Data provider implementation to integrate the
properties service for use in scenarios.</localizedDescription>
    </dataProvider>
  </items>
</dataProviders>
```

68.2.15 Viewing Detailed Data Provider Properties Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider
```

Response:

```
<dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders" capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
rel="urn:oracle:wcps:conductor:provider:providerConnections"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections/%7Bconnection-name%7D?projection={projec
tion}" resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
rel="urn:oracle:wcps:conductor:provider:providerConnection"
capabilities="urn:oracle:restframework:read"/>
    <link
```

```
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections/%7Bconnection-name%7D/resources?projecti
on={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
rel="urn:oracle:wcps:conductor:provider:connectionResources"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections/%7Bconnection-name%7D/resources/%7Bresou
rce-name%7D?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
rel="urn:oracle:wcps:conductor:provider:connectionResource"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections/%7Bconnection-name%7D/resources/%7Bresou
rce-name%7D/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:wcps:conductor:provider:resourceMethods"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/%7Bnamespace%7D/data
Providers/%7Bprovider-name%7D/connections/%7Bconnection-name%7D/resources/%7Bresou
rce-name%7D/methods/%7Bmethod-name%7D?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
rel="urn:oracle:wcps:conductor:provider:resourceMethod"
capabilities="urn:oracle:restframework:read"/>
  </links>
  <name>oracle.PropertiesServiceProvider</name>
  <localizedName>Property Service Data Provider</localizedName>
  <localizedDescription>Data provider implementation to integrate the properties
service for use in scenarios.</localizedDescription>
  <providerConnections
resourceType="urn:oracle:wcps:conductor:provider:providerConnections">
    <links>
      <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider"
capabilities="urn:oracle:restframework:read"/>
      <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections?projection={p
rojection}" resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections"
capabilities="urn:oracle:restframework:read"/>
    </links>
    <items>
      <providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
        <links>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections?projection={p
rojection}" resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
```

```
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections"
capabilities="urn:oracle:restframework:read"/>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/metadata?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/metadata" capabilities="urn:oracle:restframework:read"/>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection" capabilities="urn:oracle:restframework:read"/>
        </links>
        <name>LocalServerConnection</name>
      </providerConnection>
    </items>
  </providerConnections>
</dataProvider>
```

68.2.16 Viewing Data Provider Connections Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections
```

Response:

```
<providerConnections
resourceType="urn:oracle:wcps:conductor:provider:providerConnections">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections?projection={p
rojection}" resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections"
capabilities="urn:oracle:restframework:read"/>
  </links>
  <items>
    <providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
      <links>
        <link
```

```
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections?projection={p
rojection}" resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections"
capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/metadata?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/metadata" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection" capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>LocalServerConnection</name>
      <connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
        <links>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection" capabilities="urn:oracle:restframework:read"/>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources" capabilities="urn:oracle:restframework:read"/>
        </links>
        <items>
          <connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources" capabilities="urn:oracle:restframework:read"/>
```

```
            <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResource" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource"
capabilities="urn:oracle:restframework:read"/>
          </links>
          <name>GetPropertiesResource</name>
        </connectionResource>
      </items>
    </connectionResources>
  </providerConnection>
 </items>
</providerConnections>
```

68.2.17 Viewing Data Provider Connection Resources Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources
```

Response:

```
<connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection" capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources" capabilities="urn:oracle:restframework:read"/>
  </links>
  <items>
    <connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
```

```
nnection/resources" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResource" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <name>GetPropertiesResource</name>
      <resourceMethods
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods">
        <links>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource"
capabilities="urn:oracle:restframework:read"/>
          <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
        </links>
        <items>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(namespace,definition,
set,property)?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(namespace,definition,
set,property)" capabilities="urn:oracle:restframework:read"/>
            </links>
```

```
            <name>GetProperty</name>

<uniqueName>GetProperty(namespace,definition,set,property)</uniqueName>
            <parameters/>
          </resourceMethod>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(definition,set,proper
ty)?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(definition,set,proper
ty)" capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>GetProperty</name>
            <uniqueName>GetProperty(definition,set,property)</uniqueName>
            <parameters/>
          </resourceMethod>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/execute()?projection={projection}
" resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/execute()"
capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>execute</name>
            <uniqueName>execute()</uniqueName>
            <parameters/>
          </resourceMethod>
```

```
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySet(definition,set)?pr
ojection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySet(definition,set)"
capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>GetPropertySet</name>
            <uniqueName>GetPropertySet(definition,set)</uniqueName>
            <parameters/>
          </resourceMethod>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySet(namespace,definiti
on,set)?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySet(namespace,definiti
on,set)" capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>GetPropertySet</name>
            <uniqueName>GetPropertySet(namespace,definition,set)</uniqueName>
            <parameters/>
          </resourceMethod>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
```

```
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySets(definition)?proje
ction={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySets(definition)"
capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>GetPropertySets</name>
            <uniqueName>GetPropertySets(definition)</uniqueName>
            <parameters/>
          </resourceMethod>
          <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
            <links>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
              <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySets(namespace,definit
ion)?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetPropertySets(namespace,definit
ion)" capabilities="urn:oracle:restframework:read"/>
            </links>
            <name>GetPropertySets</name>
            <uniqueName>GetPropertySets(namespace,definition)</uniqueName>
            <parameters/>
          </resourceMethod>
        </items>
      </resourceMethods>
    </connectionResource>
  </items>
</connectionResources>
```

68.2.18 Viewing Data Provider Connection Resources Method Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(definition,set,proper
ty)
```

Response:

```
<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods"
capabilities="urn:oracle:restframework:read"/>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(definition,set,proper
ty)?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/dataProviders/oracle.PropertiesServiceProvider/connections/LocalServerCo
nnection/resources/GetPropertiesResource/methods/GetProperty(definition,set,proper
ty)" capabilities="urn:oracle:restframework:read"/>
  </links>
  <name>GetProperty</name>
  <uniqueName>GetProperty(definition,set,property)</uniqueName>
  <localizedDescription>Retrieves a property set for the current (or specified)
namespace, property set definition name, and property set name, and property name.
If no namespace, property set definition, property set and property existsnull
will be returned.</localizedDescription>
  <parameters>
    <parameter>
      <localizedDescription>The property set definition
name.</localizedDescription>
      <name>definition</name>
      <type>java.lang.String</type>
    </parameter>
    <parameter>
      <localizedDescription>The property set name.</localizedDescription>
      <name>set</name>
      <type>java.lang.String</type>
    </parameter>
    <parameter>
      <localizedDescription>The property name.</localizedDescription>
      <name>property</name>
      <type>java.lang.String</type>
    </parameter>
  </parameters>
  <returnType>java.io.Serializable</returnType>
</resourceMethod>
```

68.2.19 Viewing Function Providers Within a Namespace Using the Conductor

Request URI:

```
GET http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders
```

Response:

```
<functionProviders
resourceType="urn:oracle:wcps:conductor:provider:functionProviders">
  <links>
    <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
  </links>
  <items>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:new?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:new"
 capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>collections</category>
      <example>${collections:new()}</example>
      <localizedDescription>Creates and returns a new
collection.</localizedDescription>
      <name>new</name>
      <parameters/>
      <returnType>java.util.Collection</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/cmisfunction:getCMISQueryForDocIDs?projection={project
ion}" resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/cmisfunction:getCMISQueryForDocIDs"
```

```
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:getCMISQueryForDocIDs(repository,ids)}</example>
      <localizedDescription>Construct a CMIS query in the form of 'IN' query
syntax that will retrieve the documents for the array of input doc
IDs.</localizedDescription>
      <name>getCMISQueryForDocIDs</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/cmisfunction:getCMISQueryForDocIDsFromFullID?
projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/cmisfunction:getCMISQueryForDocIDsFromFullID"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}</example>
      <localizedDescription>Construct a CMIS query in the form of 'IN' query
 syntax that will retrieve the documents for the array of input doc
IDs.</localizedDescription>
      <name>getCMISQueryForDocIDsFromFullID</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:filterRecsByScore?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:filterRecsByScore"
 capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>agfunction</category>

<example>${agfunction:filterRecsByScore(recommendations,cutoffScore)}</example>
```

```
      <localizedDescription>Filter out Recommendations whose score is &gt;= the
input score</localizedDescription>
      <name>filterRecsByScore</name>
      <parameters/>

<returnType>oracle.wcps.activity.agrest.spy.jaxb.Recommendations</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:getContentIDs?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:getContentIDs"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>agfunction</category>
      <example>${agfunction:getContentIDs(agResults,filterClassURN)}</example>
      <localizedDescription>Extract content IDs from input
objects</localizedDescription>
      <name>getContentIDs</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:append?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:append"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>collections</category>
      <example>${collections:append(collection,item)}</example>
      <localizedDescription>Appends</localizedDescription>
      <name>append</name>
      <parameters/>
      <returnType>java.util.Collection</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
```

```
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:merge?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:merge"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>collections</category>
      <example>${collections:merge(allCollections)}</example>
      <localizedDescription>Concatenates all provided collections in the order in
which they are provided.</localizedDescription>
      <name>merge</name>
      <parameters/>
      <returnType>java.util.Collection</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:sort?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/collections:sort"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>collections</category>
      <example>${collections:sort(list)}</example>
      <localizedDescription>Sorts the provided collection.</localizedDescription>
      <name>sort</name>
      <parameters/>
      <returnType>void</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
rel="urn:oracle:webcenter:parent"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders" capabilities="urn:oracle:restframework:read"/>
        <link
```

```
template="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:getDocumentIDs?projection={projection}"
resourceType="urn:oracle:wcps:conductor:provider:functionProvider" rel="self"
href="http://localhost:8891/wcps/api/conductor/namespaces/FeatureReview_
Namespace/functionProviders/agfunction:getDocumentIDs"
capabilities="urn:oracle:restframework:read"/>
      </links>
      <category>agfunction</category>
      <example>${agfunction:getDocumentIDs(agResults,dDocNameOnly)}</example>
      <localizedDescription>Extract content IDs from input
objects</localizedDescription>
      <name>getDocumentIDs</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
  </items>
</functionProviders>
```

68.3 Property Service REST APIs

This section includes the following subsections:

- Section 68.3.1, "Property Service Resource Index"
- Section 68.3.2, "Namespace CRUD"
- Section 68.3.3, "Property Definition CRUD"
- Section 68.3.4, "Property Set Definition CRUD"
- Section 68.3.5, "Property Set CRUD"
- Section 68.3.6, "Property CRUD"
- Section 68.3.7, "JSON Payload"
- Section 68.3.8, "Exceptions"
- Section 68.3.9, "Query Parameters"
- Section 68.3.10, "Search Expressions"

68.3.1 Property Service Resource Index

This section includes the following subsections:

- Section 68.3.1.1, "ResourceIndex"
- Section 68.3.2.1, "Create Namespace"
- Section 68.3.2.2, "Retrieve Namespace"
- Section 68.3.2.3, "Update Namespace"
- Section 68.3.2.4, "Delete Namespace"
- Section 68.3.2.5, "Retrieve all Namespaces"
- Section 68.3.2.6, "Delete all Namespaces"

68.3.1.1 ResourceIndex

`ResourceIndex`

XML

Request:

```
GET http://localhost:8891/wcps/api/property/resourceIndex
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 3595
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;
path=/wcps; HttpOnly
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:15 GMT

<?xml version="1.0" encoding="UTF-8"?>
<resourceIndex resourceType="urn:oracle:wcps:property:resourceIndex">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
      href="http://localhost:8891/wcps/api/property/namespaces"
      resourceType="urn:oracle:wcps:property:namespaces"
template="http://localhost:8891/wcps/api/property/namespaces?startIndex={startInde
x}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"
      href="http://localhost:8891/wcps/api/property/resourceIndex"
      rel="self" resourceType="urn:oracle:wcps:property:resourceIndex"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:namespace"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/
>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:propertydefinition"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtydefinitions/{definitionName}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage
}"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
```

```
      resourceType="urn:oracle:wcps:property:propertysetdefinition"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions/{setDefinitionName}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dat
aFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:propertyset"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions/{setDefinitionName}/propertysets/{propertySetName}"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:properties"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions/{setDefinitionName}/propertysets/{propertySetName}/properties"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:property"
template="http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/prope
rtysetdefinitions/{setDefinitionName}/propertysets/{propertySetName}/properties/{p
ropertyName}"/>
  </links>
</resourceIndex>
```

JSON

Request:

```
GET http://localhost:8891/wcps/api/property/resourceIndex
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;
path=/wcps; HttpOnly
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:22 GMT

{
  "resourceType" : "urn:oracle:wcps:property:resourceIndex",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces?startIndex={startIndex}&q={q}&
itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:namespaces",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
```

```
    "href" : "http://localhost:8891/wcps/api/property/namespaces"
  }, {
    "resourceType" : "urn:oracle:wcps:property:resourceIndex",
    "capabilities" : "urn:oracle:restframework:read",
    "href" : "http://localhost:8891/wcps/api/property/resourceIndex",
    "rel" : "self"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertydefini
tions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertydefini
tions/{definitionName}",
    "resourceType" : "urn:oracle:wcps:property:propertydefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions/{setDefinitionName}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions/{setDefinitionName}/propertysets?startIndex={startIndex}&dataFields={data
Fields}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions/{setDefinitionName}/propertysets/{propertySetName}",
    "resourceType" : "urn:oracle:wcps:property:propertyset",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions/{setDefinitionName}/propertysets/{propertySetName}/properties",
    "resourceType" : "urn:oracle:wcps:property:properties",
```

```
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/{namespaceName}/propertysetdef
initions/{setDefinitionName}/propertysets/{propertySetName}/properties/{propertyNa
me}",
    "resourceType" : "urn:oracle:wcps:property:property",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
  } ]
}
```

68.3.2 Namespace CRUD

This section contains the following subsection:

- Section 68.3.2.1, "Create Namespace"
- Section 68.3.2.2, "Retrieve Namespace"
- Section 68.3.2.3, "Update Namespace"
- Section 68.3.2.4, "Delete Namespace"
- Section 68.3.2.5, "Retrieve all Namespaces"
- Section 68.3.2.6, "Delete all Namespaces"

68.3.2.1 Create Namespace

`CreateNamespace`

XML

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:property:namespace">
  <name>Oracle</name>
</namespace>
```

Response:

```
Status Code:201
Content-Length: 1671
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location: http://localhost:8891/wcps/api/property/namespaces/Oracle
Date: Thu, 07 Oct 2010 23:31:16 GMT
```

```
ETag: "1286494276437"

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:property:namespace">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
      href="http://localhost:8891/wcps/api/property/namespaces/Oracle"
      rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions"
      resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s"
      resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefini
tions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"
      resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={
dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <name>Oracle</name>
  <createdOn>2010-10-07T17:31:16.437-0600</createdOn>
  <updatedOn>2010-10-07T17:31:16.437-0600</updatedOn>
</namespace>
```

JSON

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:property:namespace"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
```

```
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location: http://localhost:8891/wcps/api/property/namespaces/Oracle
Date: Fri, 08 Oct 2010 01:24:26 GMT
ETag: "1286501066698"

{
  "name" : "Oracle",
  "createdOn" : "2010-10-07T19:24:26.698-0600",
  "updatedOn" : "2010-10-07T19:24:26.698-0600",
  "resourceType" : "urn:oracle:wcps:property:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle",
    "rel" : "self"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions?
startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions?sta
rtIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
{setDefinitionName}/propertysets?startIndex={startIndex}&dataFields={dataFields}&q
={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
    "capabilities" : "urn:oracle:restframework:read"
  } ]
}
```

68.3.2.2 Retrieve Namespace

`RetrieveNamespace`

XML

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
```

```
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 1759
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:16 GMT
ETag: "1286494276833"

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:property:namespace">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
      href="http://localhost:8891/wcps/api/property/namespaces/Oracle"
      rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions"
      resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s"
      resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefini
tions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"
      resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={
dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <name>Oracle</name>
  <createdOn>2010-10-07T17:31:16.437-0600</createdOn>
  <updatedOn>2010-10-07T17:31:16.833-0600</updatedOn>

<propertyLocatorClassName>com.oracle.custom.MyPropertyLocator</propertyLocatorClas
sName>
</namespace>
```

JSON

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
```

```
=1
Content-Type: application/json
Accept: application/json

Response:
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:26 GMT
ETag: "1286501066808"

{
  "name" : "Oracle",
  "createdOn" : "2010-10-07T19:24:26.698-0600",
  "propertyLocatorClassName" : "com.oracle.custom.MyPropertyLocator",
  "updatedOn" : "2010-10-07T19:24:26.808-0600",
  "resourceType" : "urn:oracle:wcps:property:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle",
    "rel" : "self"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions?
startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions?sta
rtIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
{setDefinitionName}/propertysets?startIndex={startIndex}&dataFields={dataFields}&q
={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
    "capabilities" : "urn:oracle:restframework:read"
  } ]
}
```

68.3.2.3 Update Namespace

`UpdateNamespace`

XML

Request:

```
PUT http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494276437

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:property:namespace">
  <name>Oracle</name>
  <createdOn>2010-10-07T17:31:16.437-0600</createdOn>
  <updatedOn>2010-10-07T17:31:16.437-0600</updatedOn>

<propertyLocatorClassName>com.oracle.custom.MyPropertyLocator</propertyLocatorClas
sName>
</namespace>
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:16 GMT
ETag: "1286494276833"
```

JSON

Request:

```
PUT http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501066698

{
  "name" : "Oracle",
  "propertyLocatorClassName" : "com.oracle.custom.MyPropertyLocator",
  "createdOn" : "2010-10-07T19:24:26.698-0600",
  "updatedOn" : "2010-10-07T19:24:26.698-0600",
  "resourceType" : "urn:oracle:wcps:property:namespace"
}
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
```

```
Date: Fri, 08 Oct 2010 01:24:26 GMT
ETag: "1286501066808"
```

68.3.2.4 Delete Namespace

`DeleteNamespace`

XML

Request:

```
DELETE http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494276833
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:17 GMT
```

JSON

Request:

```
DELETE http://localhost:8891/wcps/api/property/namespaces/Oracle
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501066808
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:26 GMT
```

68.3.2.5 Retrieve all Namespaces

`RetrieveAllNamespaces`

XML

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces
```

```
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:19 GMT

<?xml version="1.0" encoding="UTF-8"?>
<namespaces resourceType="urn:oracle:wcps:property:namespaces">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"
      href="http://localhost:8891/wcps/api/property/namespaces"
      rel="self" resourceType="urn:oracle:wcps:property:namespaces"
template="http://localhost:8891/wcps/api/property/namespaces?startIndex={startInde
x}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces?startIndex=10&amp;itemsPe
rPage=10"
      rel="next" resourceType="urn:oracle:wcps:property:namespaces"/>
  </links>
  <itemsPerPage>10</itemsPerPage>
  <startIndex>0</startIndex>
  <totalCount>21</totalCount>
  <items>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/JDEV"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitio
ns"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefin
itions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
```

```
href="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertydefiniti
ons?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefin
itions/{setDefinitionName}/propertysets?startIndex={startIndex}&amp;dataFields={da
taFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>JDEV</name>
      <createdOn>2010-10-07T17:31:20.061-0600</createdOn>
      <updatedOn>2010-10-07T17:31:20.061-0600</updatedOn>

<propertyLocatorClassName>oracle.wcps.conductor.scenario.persistence.impl.properti
es.MDSScenarioPropertyLocator</propertyLocatorClassName>

<definitionLocatorClassName>oracle.wcps.property.persistence.mds.MDSDefinitionsLoc
ator</definitionLocatorClassName>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_0"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_0</name>
      <createdOn>2010-10-07T17:31:17.318-0600</createdOn>
      <updatedOn>2010-10-07T17:31:17.318-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
```

```
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_1"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_1</name>
      <createdOn>2010-10-07T17:31:17.568-0600</createdOn>
      <updatedOn>2010-10-07T17:31:17.568-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_2"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
```

```
      </links>
      <name>Oracle_2</name>
      <createdOn>2010-10-07T17:31:17.807-0600</createdOn>
      <updatedOn>2010-10-07T17:31:17.807-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_3"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_3</name>
      <createdOn>2010-10-07T17:31:18.066-0600</createdOn>
      <updatedOn>2010-10-07T17:31:18.066-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_4"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
```

```
4/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_4</name>
      <createdOn>2010-10-07T17:31:18.319-0600</createdOn>
      <updatedOn>2010-10-07T17:31:18.319-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_5"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_5</name>
      <createdOn>2010-10-07T17:31:18.572-0600</createdOn>
      <updatedOn>2010-10-07T17:31:18.572-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_6"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
```

```
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_6</name>
      <createdOn>2010-10-07T17:31:18.813-0600</createdOn>
      <updatedOn>2010-10-07T17:31:18.813-0600</updatedOn>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_7"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_7</name>
      <createdOn>2010-10-07T17:31:19.084-0600</createdOn>
      <updatedOn>2010-10-07T17:31:19.084-0600</updatedOn>
    </namespace>
```

```
    <namespace resourceType="urn:oracle:wcps:property:namespace">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_8"
          rel="self" resourceType="urn:oracle:wcps:property:namespace"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions"
          resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={items
PerPage}"/>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete"
          href="http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertydefinitions"
          resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertydefinitions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPer
Page}"/>
        <link capabilities="urn:oracle:restframework:read"
          resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
amp;dataFields={dataFields}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
      </links>
      <name>Oracle_8</name>
      <createdOn>2010-10-07T17:31:19.328-0600</createdOn>
      <updatedOn>2010-10-07T17:31:19.328-0600</updatedOn>
    </namespace>
  </items>
</namespaces>
```

JSON

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:27 GMT

{
```

```
  "namespaces" : [ {
    "name" : "JDEV",
    "createdOn" : "2010-10-07T19:24:28.114-0600",
    "propertyLocatorClassName" :
"oracle.wcps.conductor.scenario.persistence.impl.properties.MDSScenarioPropertyLoc
ator",
    "definitionLocatorClassName" :
"oracle.wcps.property.persistence.mds.MDSDefinitionsLocator",
    "updatedOn" : "2010-10-07T19:24:28.114-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/JDEV",
      "rel" : "self"
    }, {
      "template" :
"http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions?st
artIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions"
    }, {
      "template" :
"http://localhost:8891/wcps/api/property/namespaces/JDEV/propertydefinitions?start
Index={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/JDEV/propertydefinitions"
    }, {
      "template" :
"http://localhost:8891/wcps/api/property/namespaces/JDEV/propertysetdefinitions/{s
etDefinitionName}/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={
q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_0",
    "createdOn" : "2010-10-07T19:24:26.907-0600",
    "updatedOn" : "2010-10-07T19:24:26.907-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_0",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
```

```
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
0/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_1",
    "createdOn" : "2010-10-07T19:24:26.954-0600",
    "updatedOn" : "2010-10-07T19:24:26.954-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_1",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
1/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_2",
    "createdOn" : "2010-10-07T19:24:27.034-0600",
    "updatedOn" : "2010-10-07T19:24:27.034-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
```

```
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_2",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
2/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_3",
    "createdOn" : "2010-10-07T19:24:27.191-0600",
    "updatedOn" : "2010-10-07T19:24:27.191-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_3",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
3/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
```

```
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_4",
    "createdOn" : "2010-10-07T19:24:27.259-0600",
    "updatedOn" : "2010-10-07T19:24:27.259-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_4",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
4/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_5",
    "createdOn" : "2010-10-07T19:24:27.359-0600",
    "updatedOn" : "2010-10-07T19:24:27.359-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_5",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
```

```
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
5/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_6",
    "createdOn" : "2010-10-07T19:24:27.409-0600",
    "updatedOn" : "2010-10-07T19:24:27.409-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_6",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertydefinitions"
    }, {
      "template" : "http:://localhost:8891/wcps/api/property/namespaces/Oracle_
6/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_7",
    "createdOn" : "2010-10-07T19:24:27.463-0600",
    "updatedOn" : "2010-10-07T19:24:27.463-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_7",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
```

```
7/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
7/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  }, {
    "name" : "Oracle_8",
    "createdOn" : "2010-10-07T19:24:27.543-0600",
    "updatedOn" : "2010-10-07T19:24:27.543-0600",
    "resourceType" : "urn:oracle:wcps:property:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_8",
      "rel" : "self"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}
",
      "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertydefinitions?startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
      "href" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertydefinitions"
    }, {
      "template" : "http://localhost:8891/wcps/api/property/namespaces/Oracle_
8/propertysetdefinitions/{setDefinitionName}/propertysets?startIndex={startIndex}&
dataFields={dataFields}&q={q}&itemsPerPage={itemsPerPage}",
      "resourceType" : "urn:oracle:wcps:property:propertysets",
      "capabilities" : "urn:oracle:restframework:read"
    } ]
  } ],
  "totalCount" : 21,
  "startIndex" : 0,
```

```
  "itemsPerPage" : 10,
  "resourceType" : "urn:oracle:wcps:property:namespaces",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces?startIndex={startIndex}&q={q}&
itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:namespaces",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" : "http://localhost:8891/wcps/api/property/namespaces",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:property:namespaces",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces?startIndex=10&itemsPerPage=10"
,
    "rel" : "next"
  } ]
}
```

68.3.2.6 Delete all Namespaces

`DeleteAllNamespaces`

XML

Request:

```
DELETE http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=t0FKMnYDSzjRlshhs0sFX20nvnTZLQY8vnxNLkplyfYPcWgfb9y6!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:17 GMT
```

JSON

Request:

```
DELETE http://localhost:8891/wcps/api/property/namespaces
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=WFGvMnyKP7nDBW1RQynhhphbQvWLDjLSbFJj2MQ1g6nFH2VQlgJ4!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:204
```

```
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:28 GMT
```

68.3.3 Property Definition CRUD

This section contains the following subsections:

- Section 68.3.3.1, "Create Property Definition - Create Integer Property Definition"
- Section 68.3.3.2, "Create Property Definition - Create Integer Array Property Definition"
- Section 68.3.3.3, "Retrieve Property Definition"
- Section 68.3.3.4, "Update Property Definition"
- Section 68.3.3.5, "Delete Property Definition"
- Section 68.3.3.6, "Retrieve all Property Definitions"
- Section 68.3.3.7, "Delete all Property Definitions"

68.3.3.1 Create Property Definition - Create Integer Property Definition

`CreateIntPD`

XML

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Def</name>
  <type>INT</type>
  <restricted>false</restricted>
  <defaultValues/>
</propertyDefinition>
```

Response:

```
Status Code:201
Content-Length: 678
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location:
```

```
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
Date: Thu, 07 Oct 2010 23:31:29 GMT
ETag: "1286494289263"

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Def"
      rel="self" resourceType="urn:oracle:wcps:property:propertydefinition"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Def</name>
  <createdOn>2010-10-07T17:31:29.263-0600</createdOn>
  <updatedOn>2010-10-07T17:31:29.263-0600</updatedOn>
  <type>INT</type>
  <restricted>false</restricted>
  <defaultValues/>
</propertyDefinition>
```

JSON

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "Integer_Def",
  "type" : "INT",
  "namespaceName" : "Oracle",
  "restricted" : false,
  "defaultValues" : [ ],
  "resourceType" : "urn:oracle:wcps:property:propertydefinition"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
Date: Fri, 08 Oct 2010 01:24:41 GMT
ETag: "1286501081086"

{
```

```
  "name" : "Integer_Def",
  "type" : "INT",
  "namespaceName" : "Oracle",
  "defaultValues" : [ ],
  "createdOn" : "2010-10-07T19:24:41.086-0600",
  "updatedOn" : "2010-10-07T19:24:41.086-0600",
  "restricted" : false,
  "resourceType" : "urn:oracle:wcps:property:propertydefinition",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:propertydefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Def",
    "rel" : "self"
  } ]
}
```

68.3.3.2 Create Property Definition - Create Integer Array Property Definition

`CreateIntArrPD`

XML

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Array_Def</name>
  <type>INT_ARRAY</type>
  <restricted>true</restricted>
  <restrictedValues>
    <integer>1</integer>
    <integer>2</integer>
    <integer>3</integer>
    <integer>4</integer>
    <integer>5</integer>
  </restrictedValues>
  <defaultValues>
    <integer>1</integer>
    <integer>2</integer>
  </defaultValues>
</propertyDefinition>
```

Response:

```
Status Code:201
Content-Length: 887
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
```

```
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Array_Def
Date: Thu, 07 Oct 2010 23:31:29 GMT
ETag: "1286494289494"

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Array_Def"
      rel="self" resourceType="urn:oracle:wcps:property:propertydefinition"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Array_Def</name>
  <createdOn>2010-10-07T17:31:29.494-0600</createdOn>
  <updatedOn>2010-10-07T17:31:29.494-0600</updatedOn>
  <type>INT_ARRAY</type>
  <restricted>true</restricted>
  <restrictedValues>
    <integer>1</integer>
    <integer>2</integer>
    <integer>3</integer>
    <integer>4</integer>
    <integer>5</integer>
  </restrictedValues>
  <defaultValues>
    <integer>1</integer>
    <integer>2</integer>
  </defaultValues>
</propertyDefinition>
```

JSON

Request:

```
POST http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "Integer_Array_Def",
  "type" : "INT_ARRAY",
  "namespaceName" : "Oracle",
  "restricted" : true,
  "restrictedValues" : [ {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 1
  }, {
    "concreteType" :
```

```
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 2
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 3
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 4
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 5
  } ],
  "defaultValues" : [ {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 1
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 2
  } ],
  "resourceType" : "urn:oracle:wcps:property:propertydefinition"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Array_Def
Date: Fri, 08 Oct 2010 01:24:41 GMT
ETag: "1286501081123"

{
  "name" : "Integer_Array_Def",
  "type" : "INT_ARRAY",
  "namespaceName" : "Oracle",
  "defaultValues" : [ {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 1
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 2
  } ],
  "createdOn" : "2010-10-07T19:24:41.123-0600",
  "updatedOn" : "2010-10-07T19:24:41.123-0600",
  "restricted" : true,
  "restrictedValues" : [ {
    "concreteType" :
```

```
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 1
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 2
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 3
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 4
  }, {
    "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
    "value" : 5
  } ],
  "resourceType" : "urn:oracle:wcps:property:propertydefinition",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:propertydefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Array_Def",
    "rel" : "self"
  } ]
}
```

68.3.3.3 Retrieve Property Definition

RetrievePD

XML

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 743
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:29 GMT
ETag: "1286494289862"
```

```
<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Def"
      rel="self" resourceType="urn:oracle:wcps:property:propertydefinition"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Def</name>
  <description>Defines an integer property value type</description>
  <createdOn>2010-10-07T17:31:29.263-0600</createdOn>
  <updatedOn>2010-10-07T17:31:29.862-0600</updatedOn>
  <type>INT</type>
  <restricted>false</restricted>
  <defaultValues/>
</propertyDefinition>
```

JSON

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:41 GMT
ETag: "1286501081172"

{
  "name" : "Integer_Def",
  "type" : "INT",
  "description" : "Defines an integer property value type",
  "namespaceName" : "Oracle",
  "defaultValues" : [ ],
  "createdOn" : "2010-10-07T19:24:41.086-0600",
  "updatedOn" : "2010-10-07T19:24:41.172-0600",
  "restricted" : false,
  "resourceType" : "urn:oracle:wcps:property:propertydefinition",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:propertydefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
```

```
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Def",
    "rel" : "self"
  } ]
}
```

68.3.3.4 Update Property Definition

UpdatePD

XML

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494289263

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinition resourceType="urn:oracle:wcps:property:propertydefinition">
  <namespaceName>Oracle</namespaceName>
  <name>Integer_Def</name>
  <description>Defines an integer property value type</description>
  <createdOn>2010-10-07T17:31:29.263-0600</createdOn>
  <updatedOn>2010-10-07T17:31:29.263-0600</updatedOn>
  <type>INT</type>
  <restricted>false</restricted>
  <defaultValues/>
</propertyDefinition>
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:29 GMT
ETag: "1286494289862"
```

JSON

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

```
If-Match: 1286501081086

{
  "name" : "Integer_Def",
  "type" : "INT",
  "description" : "Defines an integer property value type",
  "namespaceName" : "Oracle",
  "createdOn" : "2010-10-07T19:24:41.086-0600",
  "updatedOn" : "2010-10-07T19:24:41.086-0600",
  "restricted" : false,
  "defaultValues" : [ ],
  "resourceType" : "urn:oracle:wcps:property:propertydefinition"
}
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:41 GMT
ETag: "1286501081172"
```

68.3.3.5 Delete Property Definition

DeletePD

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494289263
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:29 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Inte
ger_Def
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
```

```
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501081086
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:41 GMT
```

68.3.3.6 Retrieve all Property Definitions

`RetrieveAllPDs`

XML

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 1586
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:29 GMT

<?xml version="1.0" encoding="UTF-8"?>
<propertyDefinitions resourceType="urn:oracle:wcps:property:propertydefinitions">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s?q=pd-name:like:%25&amp;startIndex=0&amp;itemsPerPage=10"
      rel="self"
      resourceType="urn:oracle:wcps:property:propertydefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefini
tions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <itemsPerPage>10</itemsPerPage>
  <startIndex>0</startIndex>
```

```
  <totalCount>1</totalCount>
  <namespaceName>Oracle</namespaceName>
  <items>
    <propertyDefinition
resourceType="urn:oracle:wcps:property:propertydefinition">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Array_Def"
          rel="self" resourceType="urn:oracle:wcps:property:propertydefinition"/>
      </links>
      <name>Integer_Array_Def</name>
      <createdOn>2010-10-07T17:31:29.494-0600</createdOn>
      <updatedOn>2010-10-07T17:31:29.494-0600</updatedOn>
      <type>INT_ARRAY</type>
      <restricted>true</restricted>
      <restrictedValues>
        <integer>5</integer>
        <integer>4</integer>
        <integer>3</integer>
        <integer>2</integer>
        <integer>1</integer>
      </restrictedValues>
      <defaultValues>
        <integer>2</integer>
        <integer>1</integer>
      </defaultValues>
    </propertyDefinition>
  </items>
</propertyDefinitions>
```

JSON

Request:

```
GET http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:41 GMT

{
  "namespaceName" : "Oracle",
  "propertyDefinitions" : [ {
    "name" : "Integer_Array_Def",
    "type" : "INT_ARRAY",
```

```
    "defaultValues" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 2
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 1
    } ],
    "createdOn" : "2010-10-07T19:24:41.123-0600",
    "updatedOn" : "2010-10-07T19:24:41.123-0600",
    "restricted" : true,
    "restrictedValues" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 5
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 4
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 3
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 2
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 1
    } ],
    "resourceType" : "urn:oracle:wcps:property:propertydefinition",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:propertydefinition",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Array_Def",
      "rel" : "self"
    } ]
  } ],
  "totalCount" : 1,
  "startIndex" : 0,
  "itemsPerPage" : 10,
  "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions?sta
rtIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertydefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions?q=p
d-name:like:%25&startIndex=0&itemsPerPage=10",
    "rel" : "self"
  } ]
```

```
}
```

68.3.3.7 Delete all Property Definitions

DeleteAllPDs

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=ZfhzMnYQz7pg4gzv5wZV1srvy3yDTsG4J4090X1GnHhrQpCSTkG2!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:30 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=GSw4MnyYTw87Q6yCBb2H9F0JL6kLJwjrqYv3b72LtDMQGdLKPGRc!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:41 GMT
```

68.3.4 Property Set Definition CRUD

This section contains the following subsections:

- Section 68.3.4.1, "Create Property Set Definition"
- Section 68.3.4.2, "Retrieve Property Set Definition"
- Section 68.3.4.3, "Retrieve All Property Set Definitions"
- Section 68.3.4.4, "Delete all Property Set Definitions"

68.3.4.1 Create Property Set Definition

`CreatePSD`

XML

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7sCPDNFlyzQGRSRnXlfHp8lSTbR8NH!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<propertySetDefinition
resourceType="urn:oracle:wcps:property:propertysetdefinition">
  <namespaceName>Oracle</namespaceName>
  <name>partnersinfo</name>
  <propertyMapping>
    <propertyName>company_name</propertyName>
    <propertyDefinitionName>String_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <propertyName>address</propertyName>
    <propertyDefinitionName>String_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <propertyName>number_of_employees</propertyName>
    <propertyDefinitionName>Integer_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <propertyName>previous_ranking_and_current_ranking</propertyName>
    <propertyDefinitionName>Integer_Array_Def</propertyDefinitionName>
  </propertyMapping>
</propertySetDefinition>
```

Response:

```
Status Code:201
Content-Length: 2854
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo
Date: Thu, 07 Oct 2010 23:31:32 GMT
ETag: "1286494292658"

<?xml version="1.0" encoding="UTF-8"?>
<propertySetDefinition
resourceType="urn:oracle:wcps:property:propertysetdefinition">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
```

```
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo"
      rel="self" resourceType="urn:oracle:wcps:property:propertysetdefinition"/>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets"
      resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/partnersinfo/propertysets?startIndex={startIndex}&amp;dataFields={dataFie
lds}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <name>partnersinfo</name>
  <createdOn>2010-10-07T17:31:32.658-0600</createdOn>
  <updatedOn>2010-10-07T17:31:32.658-0600</updatedOn>
  <propertyMapping>
    <links>
      <link
        capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Def" resourceType="urn:oracle:wcps:property:propertydefinition"/>
    </links>
    <propertyName>number_of_employees</propertyName>
    <propertyDefinitionName>Integer_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <links>
      <link
        capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/String_Def" resourceType="urn:oracle:wcps:property:propertydefinition"/>
    </links>
    <propertyName>address</propertyName>
    <propertyDefinitionName>String_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <links>
      <link
        capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/String_Def" resourceType="urn:oracle:wcps:property:propertydefinition"/>
    </links>
    <propertyName>company_name</propertyName>
    <propertyDefinitionName>String_Def</propertyDefinitionName>
  </propertyMapping>
  <propertyMapping>
    <links>
      <link
        capabilities="urn:oracle:restframework:read
```

```
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinition
s/Integer_Array_Def" resourceType="urn:oracle:wcps:property:propertydefinition"/>
    </links>
    <propertyName>previous_ranking_and_current_ranking</propertyName>
    <propertyDefinitionName>Integer_Array_Def</propertyDefinitionName>
  </propertyMapping>
</propertySetDefinition>
```

JSON

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=dpvQMnyDkSNwv1TZ20JSYyzJpKJn1JwBQdRtNLQhNPHWpgvs8PcZ!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "partnersinfo",
  "namespaceName" : "Oracle",
  "propertyMappings" : [ {
    "propertyDefinitionName" : "String_Def",
    "propertyName" : "company_name"
  }, {
    "propertyDefinitionName" : "String_Def",
    "propertyName" : "address"
  }, {
    "propertyDefinitionName" : "Integer_Def",
    "propertyName" : "number_of_employees"
  }, {
    "propertyDefinitionName" : "Integer_Array_Def",
    "propertyName" : "previous_ranking_and_current_ranking"
  } ],
  "resourceType" : "urn:oracle:wcps:property:propertysetdefinition"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo
Date: Fri, 08 Oct 2010 01:24:51 GMT
ETag: "1286501091845"

{
  "name" : "partnersinfo",
  "namespaceName" : "Oracle",
  "createdOn" : "2010-10-07T19:24:51.845-0600",
  "updatedOn" : "2010-10-07T19:24:51.845-0600",
```

```
  "propertyMappings" : [ {
    "propertyName" : "number_of_employees",
    "propertyDefinitionName" : "Integer_Def",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:propertydefinition",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Def"
    } ]
  }, {
    "propertyName" : "address",
    "propertyDefinitionName" : "String_Def",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:propertydefinition",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Str
ing_Def"
    } ]
  }, {
    "propertyName" : "company_name",
    "propertyDefinitionName" : "String_Def",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:propertydefinition",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Str
ing_Def"
    } ]
  }, {
    "propertyName" : "previous_ranking_and_current_ranking",
    "propertyDefinitionName" : "Integer_Array_Def",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:propertydefinition",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertydefinitions/Int
eger_Array_Def"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:property:propertysetdefinition",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinition",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo",
    "rel" : "self"
  }, {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={q}&it
emsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
```

```
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets"
  } ]
}
```

68.3.4.2 Retrieve Property Set Definition

`RetrievePSD`

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7sCPDNFlyzQGRSRnXlfHp8lSTbR8NH!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494292658
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:33 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=dpvQMnyDkSNwv1TZ20JSYyzJpKJn1JwBQdRtNLQhNPHWpgvs8PcZ!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501091845
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:51 GMT
```

68.3.4.3 Retrieve All Property Set Definitions

RetrieveAllPSDs

XML

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7sCPDNFlyzQGRSRnXlfHp8lSTbR8NH!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 779
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:33 GMT

<?xml version="1.0" encoding="UTF-8"?>
<propertySetDefinitions
resourceType="urn:oracle:wcps:property:propertysetdefinitions">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions?startIndex=0&amp;itemsPerPage=10"
      rel="self"
      resourceType="urn:oracle:wcps:property:propertysetdefinitions"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions?startIndex={startIndex}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <itemsPerPage>10</itemsPerPage>
  <startIndex>0</startIndex>
  <totalCount>0</totalCount>
  <namespaceName>Oracle</namespaceName>
  <items/>
</propertySetDefinitions>
```

JSON

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=dpvQMnyDkSNwv1TZ20JSYyzJpKJn1JwBQdRtNLQhNPHWpgvs8PcZ!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:51 GMT

{
  "namespaceName" : "Oracle",
  "propertySetDefinitions" : [ ],
  "totalCount" : 0,
  "startIndex" : 0,
  "itemsPerPage" : 10,
  "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions?
startIndex={startIndex}&q={q}&itemsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysetdefinitions",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions?
startIndex=0&itemsPerPage=10",
    "rel" : "self"
  } ]
}
```

68.3.4.4 Delete all Property Set Definitions

`DeleteAllPSDs`

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=6Q93MnYT5xcTQJ3tPLn3vT7sCPDNFlyzQGRSRnXlfHp8lSTbR8NH!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:33 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=dpvQMnyDkSNwv1TZ20JSYyzJpKJn1JwBQdRtNLQhNPHWpgvs8PcZ!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:51 GMT
```

68.3.5 Property Set CRUD

This section contains the following subsections:

- Section 68.3.5.1, "Create Property Set"
- Section 68.3.5.2, "Retrieve Property Set"
- Section 68.3.5.3, "Update Property Set"
- Section 68.3.5.4, "Delete Property Set"
- Section 68.3.5.5, "Retrieve all Property Sets"
- Section 68.3.5.6, "Delete all Property Sets"

68.3.5.1 Create Property Set

`CreatePS`

XML

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<propertySet resourceType="urn:oracle:wcps:property:propertyset">
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <name>infosys</name>
  <items>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
```

```
      <name>company_name</name>
      <type>STRING</type>
      <values>
        <string>InfoSys</string>
      </values>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <name>address</name>
      <type>STRING</type>
      <values>
        <string>1 foo loop, San Francisco, CA</string>
      </values>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <name>number_of_employees</name>
      <type>INT</type>
      <values>
        <integer>10000</integer>
      </values>
    </property>
    <property multiValue="true" resourceType="urn:oracle:wcps:property:property">
      <name>previous_ranking_and_current_ranking</name>
      <type>INT_ARRAY</type>
      <values>
        <integer>3</integer>
        <integer>5</integer>
      </values>
    </property>
  </items>
</propertySet>

Response:

Status Code:201
Content-Length: 3709
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
Date: Thu, 07 Oct 2010 23:31:35 GMT
ETag: "1286494295948"

<?xml version="1.0" encoding="UTF-8"?>
<propertySet resourceType="urn:oracle:wcps:property:propertyset">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:property"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/partnersinfo/propertysets/infosys/properties/{propertyName}"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
```

```
href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys"
      rel="self" resourceType="urn:oracle:wcps:property:propertyset"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <name>infosys</name>
  <items>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/number_of_employees"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>number_of_employees</name>
      <type>INT</type>
      <values>
        <integer>10000</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/address"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>address</name>
      <type>STRING</type>
      <values>
        <string>1 foo loop, San Francisco, CA</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/company_name"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>company_name</name>
      <type>STRING</type>
      <values>
```

```
        <string>InfoSys</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="true" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_
ranking"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>previous_ranking_and_current_ranking</name>
      <type>INT_ARRAY</type>
      <values>
        <integer>3</integer>
        <integer>5</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
  </items>
  <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
  <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
</propertySet>
```

JSON

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "infosys",
  "properties" : [ {
    "name" : "company_name",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "InfoSys"
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "address",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
```

```
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "1 foo loop, San Francisco, CA"
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "number_of_employees",
    "type" : "INT",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 10000
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "previous_ranking_and_current_ranking",
    "type" : "INT_ARRAY",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 3
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 5
    } ],
    "multiValue" : true,
    "resourceType" : "urn:oracle:wcps:property:property"
  } ],
  "namespaceName" : "Oracle",
  "propertySetDefinitionName" : "partnersinfo",
  "resourceType" : "urn:oracle:wcps:property:propertyset"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
Date: Fri, 08 Oct 2010 01:24:58 GMT
ETag: "1286501098924"

{
  "name" : "infosys",
  "properties" : [ {
    "name" : "number_of_employees",
    "type" : "INT",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 10000
    } ],
```

```
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/number_of_employees",
      "rel" : "self"
    } ]
  }, {
    "name" : "address",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "1 foo loop, San Francisco, CA"
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/address",
      "rel" : "self"
    } ]
  }, {
    "name" : "company_name",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "InfoSys"
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/company_name",
      "rel" : "self"
    } ]
  }, {
    "name" : "previous_ranking_and_current_ranking",
```

```
    "type" : "INT_ARRAY",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 3
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 5
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "multiValue" : true,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_
ranking",
      "rel" : "self"
    } ]
  } ],
  "namespaceName" : "Oracle",
  "createdOn" : "2010-10-07T19:24:58.924-0600",
  "updatedOn" : "2010-10-07T19:24:58.924-0600",
  "propertySetDefinitionName" : "partnersinfo",
  "resourceType" : "urn:oracle:wcps:property:propertyset",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/{propertyName}",
    "resourceType" : "urn:oracle:wcps:property:property",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
  }, {
    "resourceType" : "urn:oracle:wcps:property:propertyset",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys",
    "rel" : "self"
  } ]
}
```

68.3.5.2 Retrieve Property Set

`RetrievePS`

XML

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
```

```
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 3709
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:36 GMT
ETag: "1286494296325"

<?xml version="1.0" encoding="UTF-8"?>
<propertySet resourceType="urn:oracle:wcps:property:propertyset">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"
      resourceType="urn:oracle:wcps:property:property"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/partnersinfo/propertysets/infosys/properties/{propertyName}"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys"
      rel="self" resourceType="urn:oracle:wcps:property:propertyset"/>
  </links>
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <name>infosys</name>
  <items>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/number_of_employees"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>number_of_employees</name>
      <type>INT</type>
      <values>
        <integer>10000</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:36.325-0600</updatedOn>
    </property>
```

```
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/company_name"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>company_name</name>
      <type>STRING</type>
      <values>
        <string>InfoSys</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:36.325-0600</updatedOn>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/address"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>address</name>
      <type>STRING</type>
      <values>
        <string>1 foo loop, San Francisco, CA</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:36.325-0600</updatedOn>
    </property>
    <property multiValue="true" resourceType="urn:oracle:wcps:property:property">
      <links>
        <link
          capabilities="urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_
ranking"
          rel="self" resourceType="urn:oracle:wcps:property:property"/>
      </links>
      <name>previous_ranking_and_current_ranking</name>
      <type>INT_ARRAY</type>
      <values>
        <integer>4</integer>
        <integer>5</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:36.325-0600</updatedOn>
    </property>
  </items>
```

```
  <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
  <updatedOn>2010-10-07T17:31:36.325-0600</updatedOn>
</propertySet>
```

JSON

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:58 GMT
ETag: "1286501098964"

{
  "name" : "infosys",
  "properties" : [ {
    "name" : "number_of_employees",
    "type" : "INT",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 10000
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.964-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/number_of_employees",
      "rel" : "self"
    } ]
  }, {
    "name" : "company_name",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "InfoSys"
    } ],
```

```
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.964-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/company_name",
      "rel" : "self"
    } ]
  }, {
    "name" : "address",
    "type" : "STRING",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "1 foo loop, San Francisco, CA"
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.964-0600",
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/address",
      "rel" : "self"
    } ]
  }, {
    "name" : "previous_ranking_and_current_ranking",
    "type" : "INT_ARRAY",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 4
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 5
    } ],
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.964-0600",
    "multiValue" : true,
    "resourceType" : "urn:oracle:wcps:property:property",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:property:property",
      "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
      "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/previous_ranking_and_current_
```

```
ranking",
      "rel" : "self"
    } ]
  } ],
  "namespaceName" : "Oracle",
  "createdOn" : "2010-10-07T19:24:58.924-0600",
  "updatedOn" : "2010-10-07T19:24:58.964-0600",
  "propertySetDefinitionName" : "partnersinfo",
  "resourceType" : "urn:oracle:wcps:property:propertyset",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/{propertyName}",
    "resourceType" : "urn:oracle:wcps:property:property",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete"
  }, {
    "resourceType" : "urn:oracle:wcps:property:propertyset",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys",
    "rel" : "self"
  } ]
}
```

68.3.5.3 Update Property Set

`UpdatePS`

XML

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494295948

<?xml version="1.0" encoding="UTF-8"?>
<propertySet resourceType="urn:oracle:wcps:property:propertyset">
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <name>infosys</name>
  <items>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <name>number_of_employees</name>
      <type>INT</type>
      <values>
        <integer>10000</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
```

```
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <name>address</name>
      <type>STRING</type>
      <values>
        <string>1 foo loop, San Francisco, CA</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="false" resourceType="urn:oracle:wcps:property:property">
      <name>company_name</name>
      <type>STRING</type>
      <values>
        <string>InfoSys</string>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
    <property multiValue="true" resourceType="urn:oracle:wcps:property:property">
      <name>previous_ranking_and_current_ranking</name>
      <type>INT_ARRAY</type>
      <values>
        <integer>4</integer>
        <integer>5</integer>
      </values>
      <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
      <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
    </property>
  </items>
  <createdOn>2010-10-07T17:31:35.948-0600</createdOn>
  <updatedOn>2010-10-07T17:31:35.948-0600</updatedOn>
</propertySet>
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:36 GMT
ETag: "1286494296325"
```

JSON

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501098924

{
```

```
  "name" : "infosys",
  "properties" : [ {
    "name" : "number_of_employees",
    "type" : "INT",
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 10000
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "address",
    "type" : "STRING",
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "1 foo loop, San Francisco, CA"
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "company_name",
    "type" : "STRING",
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:string-value",
      "value" : "InfoSys"
    } ],
    "multiValue" : false,
    "resourceType" : "urn:oracle:wcps:property:property"
  }, {
    "name" : "previous_ranking_and_current_ranking",
    "type" : "INT_ARRAY",
    "createdOn" : "2010-10-07T19:24:58.924-0600",
    "updatedOn" : "2010-10-07T19:24:58.924-0600",
    "values" : [ {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 4
    }, {
      "concreteType" :
"urn:oracle:wcps:property:internal:jaxrs:model:integer-value",
      "value" : 5
    } ],
    "multiValue" : true,
    "resourceType" : "urn:oracle:wcps:property:property"
  } ],
  "namespaceName" : "Oracle",
  "createdOn" : "2010-10-07T19:24:58.924-0600",
  "updatedOn" : "2010-10-07T19:24:58.924-0600",
  "propertySetDefinitionName" : "partnersinfo",
  "resourceType" : "urn:oracle:wcps:property:propertyset"
}
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:58 GMT
ETag: "1286501098964"
```

68.3.5.4 Delete Property Set

DeletePS

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494295948
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:36 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501098924
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
```

```
Date: Fri, 08 Oct 2010 01:24:58 GMT
```

68.3.5.5 Retrieve all Property Sets

`RetrieveAllPSs`

XML

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 886
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:36 GMT

<?xml version="1.0" encoding="UTF-8"?>
<propertySets resourceType="urn:oracle:wcps:property:propertysets">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets?startIndex=0&amp;itemsPerPage=10"
      rel="self" resourceType="urn:oracle:wcps:property:propertysets"
template="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdef
initions/partnersinfo/propertysets?startIndex={startIndex}&amp;dataFields={dataFie
lds}&amp;q={q}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
  <itemsPerPage>10</itemsPerPage>
  <startIndex>0</startIndex>
  <totalCount>0</totalCount>
  <namespaceName>Oracle</namespaceName>
  <propertySetDefinitionName>partnersinfo</propertySetDefinitionName>
  <items/>
</propertySets>
```

JSON

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
```

```
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:59 GMT

{
  "propertySets" : [ ],
  "namespaceName" : "Oracle",
  "propertySetDefinitionName" : "partnersinfo",
  "totalCount" : 0,
  "startIndex" : 0,
  "itemsPerPage" : 10,
  "resourceType" : "urn:oracle:wcps:property:propertysets",
  "links" : [ {
    "template" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets?startIndex={startIndex}&dataFields={dataFields}&q={q}&it
emsPerPage={itemsPerPage}",
    "resourceType" : "urn:oracle:wcps:property:propertysets",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets?startIndex=0&itemsPerPage=10",
    "rel" : "self"
  } ]
}
```

68.3.5.6 Delete all Property Sets

DeleteAllPSs

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
```

```
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Thu, 07 Oct 2010 23:31:36 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:59 GMT
```

68.3.6 Property CRUD

This section contains the following subsections:

- Section 68.3.6.1, "Create Property"
- Section 68.3.6.2, "Retrieve Property"
- Section 68.3.6.3, "Update Property"
- Section 68.3.6.4, "Delete Property"
- Section 68.3.6.5, "Delete all Properties"

68.3.6.1 Create Property

`CreateP`

XML

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<property multiValue="false" resourceType="urn:oracle:wcps:property:property">
  <name>ceo_name</name>
  <type>STRING</type>
  <values>
    <string>Mr. Smith</string>
  </values>
</property>
```

Response:

```
Status Code:201
Content-Length: 694
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
Date: Thu, 07 Oct 2010 23:31:36 GMT
ETag: "1286494297011"

<?xml version="1.0" encoding="UTF-8"?>
<property multiValue="false" resourceType="urn:oracle:wcps:property:property">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/ceo_name"
      rel="self" resourceType="urn:oracle:wcps:property:property"/>
  </links>
  <name>ceo_name</name>
  <type>STRING</type>
  <values>
    <string>Mr. Smith</string>
  </values>
  <createdOn>2010-10-07T17:31:36.741-0600</createdOn>
  <updatedOn>2010-10-07T17:31:37.011-0600</updatedOn>
</property>
```

JSON

Request:

```
POST
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json

{
  "name" : "ceo_name",
  "type" : "STRING",
  "values" : [ {
```

```
    "concreteType" : "urn:oracle:wcps:property:internal:jaxrs:model:string-value",
    "value" : "Mr. Smith"
  } ],
  "multiValue" : false,
  "resourceType" : "urn:oracle:wcps:property:property"
}
```

Response:

```
Status Code:201
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
Date: Fri, 08 Oct 2010 01:24:59 GMT
ETag: "1286501099056"

{
  "name" : "ceo_name",
  "type" : "STRING",
  "values" : [ {
    "concreteType" : "urn:oracle:wcps:property:internal:jaxrs:model:string-value",
    "value" : "Mr. Smith"
  } ],
  "createdOn" : "2010-10-07T19:24:59.038-0600",
  "updatedOn" : "2010-10-07T19:24:59.056-0600",
  "multiValue" : false,
  "resourceType" : "urn:oracle:wcps:property:property",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:property",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/ceo_name",
    "rel" : "self"
  } ]
}
```

68.3.6.2 Retrieve Property

RetrieveP

XML

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
```

```
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 694
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Content-Type: application/xml
Date: Thu, 07 Oct 2010 23:31:37 GMT
ETag: "1286494297366"

<?xml version="1.0" encoding="UTF-8"?>
<property multiValue="false" resourceType="urn:oracle:wcps:property:property">
  <links>
    <link
      capabilities="urn:oracle:restframework:create urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete"

href="http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinit
ions/partnersinfo/propertysets/infosys/properties/ceo_name"
      rel="self" resourceType="urn:oracle:wcps:property:property"/>
  </links>
  <name>ceo_name</name>
  <type>STRING</type>
  <values>
    <string>Mr. Baker</string>
  </values>
  <createdOn>2010-10-07T17:31:36.741-0600</createdOn>
  <updatedOn>2010-10-07T17:31:37.366-0600</updatedOn>
</property>
```

JSON

Request:

```
GET
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:200
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Transfer-Encoding: chunked
Content-Type: application/json
Date: Fri, 08 Oct 2010 01:24:59 GMT
ETag: "1286501099162"

{
```

```
  "name" : "ceo_name",
  "type" : "STRING",
  "values" : [ {
    "concreteType" : "urn:oracle:wcps:property:internal:jaxrs:model:string-value",
    "value" : "Mr. Baker"
  } ],
  "createdOn" : "2010-10-07T19:24:59.038-0600",
  "updatedOn" : "2010-10-07T19:24:59.162-0600",
  "multiValue" : false,
  "resourceType" : "urn:oracle:wcps:property:property",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:property:property",
    "capabilities" : "urn:oracle:restframework:create
urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete",
    "href" :
"http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/
partnersinfo/propertysets/infosys/properties/ceo_name",
    "rel" : "self"
  } ]
}
```

68.3.6.3 Update Property

UpdateP

XML

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
Cookie:
JSESSIONID=vc1pMnYWRZysgqhXVz5XrGKqhvfy0vv2zc7xQ3RyDSvG0mG22ZlJ!-1714998420;Versio
n=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286494297011

<?xml version="1.0" encoding="UTF-8"?>
<property multiValue="false" resourceType="urn:oracle:wcps:property:property">
  <name>ceo_name</name>
  <type>STRING</type>
  <values>
    <string>Mr. Baker</string>
  </values>
  <createdOn>2010-10-07T17:31:36.741-0600</createdOn>
  <updatedOn>2010-10-07T17:31:37.011-0600</updatedOn>
</property>
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FC59cJkhnJPolzPwKsStfD0Dxjm8_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
```

```
Date: Thu, 07 Oct 2010 23:31:37 GMT
ETag: "1286494297366"
```

JSON

Request:

```
PUT
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501099056

{
  "name" : "ceo_name",
  "type" : "STRING",
  "createdOn" : "2010-10-07T19:24:59.038-0600",
  "updatedOn" : "2010-10-07T19:24:59.056-0600",
  "values" : [ {
    "concreteType" : "urn:oracle:wcps:property:internal:jaxrs:model:string-value",
    "value" : "Mr. Baker"
  } ],
  "multiValue" : false,
  "resourceType" : "urn:oracle:wcps:property:property"
}
```

Response:

```
Status Code:200
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Cache-Control: no-cache
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:59 GMT
ETag: "1286501099162"
```

68.3.6.4 Delete Property

DeleteP

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=VyThMnyG0gyp97J3vKPrkfF4VvtrjTDHJ3SL894V0glWzqJ24L0B!-609606431;Version
=1
Content-Type: application/xml
Accept: application/xml
If-Match: 1286501097791
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:58 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties/ceo_name
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
If-Match: 1286501099162
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:59 GMT
```

68.3.6.5 Delete all Properties

`DeleteAllP`

XML

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=VyThMnyG0gyp97J3vKPrkfF4VvtrjTDHJ3SL894V0glWzqJ24L0B!-609606431;Version
=1
Content-Type: application/xml
Accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:58 GMT
```

JSON

Request:

```
DELETE
http://localhost:8891/wcps/api/property/namespaces/Oracle/propertysetdefinitions/p
artnersinfo/propertysets/infosys/properties
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
Cookie:
JSESSIONID=h72xMnyKQQSW9Lwcg2HMGDnNQnK1HZJpGJmLgZ2P1vHFpvT2bX2k!-609606431;Version
=1
Content-Type: application/json
Accept: application/json
```

Response:

```
Status Code:204
Content-Length: 0
X-Oracle-RF-Token-Scheme: utoken
X-Oracle-RF-Token: FIxec1lmTFguxer9ZqZJpsXSnlZy_w**
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
X-Oracle-RF-Token-Location: header
Date: Fri, 08 Oct 2010 01:24:59 GMT
```

68.3.7 JSON Payload

The property value in JSON format indicates the value type using `concreteType` field names. The following table maps the property definition type and Java data type to the values of `concreteType`.

Table 68–2 concreteType Values

Property Definition Type	Java Data Type	concreteType Value
INT	java.lang.Integer	urn:oracle:wcps:property:internal:jaxrs:model:integer-value
NUMBER	java.lang.Double	urn:oracle:wcps:property:internal:jaxrs:model:double-value
STRING	java.lang.String	urn:oracle:wcps:property:internal:jaxrs:model:string-value
BOOLEAN	java.lang.Boolean	urn:oracle:wcps:property:internal:jaxrs:model:boolean-value
BLOB	oracle.wcps.property.BlobData	urn:oracle:wcps:property:blob-data
CLOB	oracle.wcps.property.ClobData	urn:oracle:wcps:property:clob-data
DATETIME	oracle.wcps.property.DateTime	urn:oracle:wcps:property:date-time

68.3.8 Exceptions

- If optimistic concurrency is disabled on the property server, the "If-Match" request header is not required. If enabled, the value of the "If-Match" header should be the value sent for the response header "ETag".
- Not sending the request header X-Oracle-RF-Token results in a 403 response status. The value of the token should be the one sent in response of the `/resouceIndex` request.
- Attempting to delete an "in use" Namespace, Property Definition, or Property Set Definition results in a 412 response status.

68.3.9 Query Parameters

Table 68–3 shows the supported query parameters and their default values.

Table 68–3 Query Parameters

Query Parameter Name	Description	Default Value	Other Special Values
q	the value for it is a search expression	none	none
startIndex	Represents the index of the first element in a page	0	none
itemsPerPage	Represents the number of items to be included in a page	10	none
dataFields	Represents a list of attributes of a property service entities for which data is returned	empty list = (Return zero attributes)	* = (all attributes)

68.3.10 Search Expressions

This section contains the following subsections:

- Section 68.3.10.1, "Format"
- Section 68.3.10.2, "Supported Operators"
- Section 68.3.10.3, "Supported Attribute Names"

68.3.10.1 Format

The value for the 'q' query parameter is a search expression. The search expression format is:

```
single_expression = attribute_name:operator_name:value
expression = single_expression[;single_expression]
```

68.3.10.2 Supported Operators

- Relational Operator
  - `:equals:` ==> Equals
  - `:contains:` ==> Contains
  - `:like:` ==> SQL Like, applies only to string type property values
  - `:gt:` ==> Greater than
  - `:gte:` ==> Greater than or equals to
  - `:lt:` ==> Less than
  - `:lte:` ==> Less than or equals to
- Logical Opertion
  - `;` ==> and

68.3.10.3 Supported Attribute Names

Table 68–4, Table 68–5, Table 68–6, and Table 68–7 show the supported attribute names that can be used in a search expressions to search for property service entities.

Namespace Attributes

Table 68–4 Namespace Attributes

Attribute Name	Description
ns-name	A namespace's name
ns-property-locator-class-name	A namespace's property locator class name
ns-definition-locator-class-name	A namespace's definition locator class name
ns-created-on	A namespace's created on date time
ns-updated-on	A namespace's updated on date time

Examples:

```
/wcps/api/property/namespaces?q=ns-name:like:Oracle_%25
```

```
/wcps/api/property/namespaces?q=ns-definition-locator-class-name:like:%25Locator%2
5;ns-name:like:Oracle_%25
```

Property Definition Attributes

Table 68–5 Property Definition Attributes

Attribute Name	Description
pd-name	A property definition's name
pd-description	A property definition's description
pd-type	A property definition's type
pd-restricted	A property definition's restricted attribute
pd-validator-class-name	A property definition's validator class name
pd-created-on	A property definition's created on date time
pd-updated-on	A property definition's updated on date time

Example:

```
/wcps/api/property/namespaces/Oracle/propertydefinitions?q=pd-name:like:%25D%25;pd
-description:contains:4th;pd-restricted:contains:true
```

Property Set Definition Attributes

Table 68–6 Property Set Definition Attributes

Attribute Name	Description
psd-name	A property set definition's name
psd-description	A property set definition's description
psd-pd-name	A property definition name referred by a property set definition
psd-property-locator-class-name	A property locator class name associated with a property set definition
psd-created-on	A property set definition's created on date time
psd-updated-on	A property set definition's updated on date time

Example:

```
wcps/api/property/namespaces/Oracle/propertysetdefinitions?q=psd-created-on:gt:198
0-01-01T00:00:00.000-0700;psd-name:like:%255%25
```

Property Set Attributes

Table 68–7 Property Set Attributes

Attribute Name	Description
ps-name	A property set's name
ps-description	A property set's description
psd-created-on	A property set's created on date time
psd-updated-on	A property set's updated on date time
a property's name	A property's name attribute (for example, firstName, zipCode, etc.)

Example:

```
/wcps/api/property/namespaces/Oracle/propertysetdefinitions/partners_
info/propertysets?q=ps-created-on:gt:1980-01-01T00:00:00.000-0700
```

68.4 Using the Conductor REST APIs

This section provides examples of how you can use the Conductor's REST APIs to programatically carry out basic Personalization tasks, such as:

- Get information about data providers
- Get information about function providers
- Create, retrieve, or delete a namespace
- Create, retrieve, execute, or delete a scenario

You can use the Conductor REST APIs from within application code, or using Personalization ELs to call them from within your WebCenter Portal: Framework application or WebCenter Portal: Spaces.

This section contains the following subsections:

- Section 68.4.1, "Using the Data Provider Management APIs"
- Section 68.4.2, "Using the Function Provider Management APIs"
- Section 68.4.3, "Using the Namespace Management APIs"
- Section 68.4.4, "Using the Scenario Management APIs"

68.4.1 Using the Data Provider Management APIs

This section provides examples of using the Conductor's data provider management APIs, and contains the following subsections:

- Section 68.4.1.1, "Getting a Collection of Data Providers (GET)"
- Section 68.4.1.2, "Getting a Single Data Provider's Metadata (GET)"
- Section 68.4.1.3, "Getting Information About a Data Provider Connection (GET)"

- Section 68.4.1.4, "Getting Information about an Executable Resource for a Data Provider and Connection (GET)"
- Section 68.4.1.5, "Getting Method Information for an Executable Resource for a Data Provider and Connection (GET)"
- Section 68.4.1.6, "Getting Entire Data Provider Information with Details Projection (GET)"

68.4.1.1 Getting a Collection of Data Providers (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projectio
n=summary
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 4469
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=YxYmM25GLpCy7BSND7vF6BnsQCn20RQsCrS92n1SLXmxpccVkfpX!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 22:58:46 GMT

<?xml version="1.0" encoding="UTF-8"?>
<dataProviders resourceType="urn:oracle:wcps:conductor:provider:dataProviders">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?pro
jection=summary&amp;startIndex=0&amp;itemsPerPage=10"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
  </links>
  <items>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/CMI
SProvider"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
```

```
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/CMISProvider?projection={projection}"/>
      </links>
      <name>CMISProvider</name>
      <localizedName>Universal Content Provider (CMIS)</localizedName>
      <localizedDescription>Provides access to content over CMIS
connections.</localizedDescription>
    </dataProvider>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/Sam
pleProvider"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/SampleProvider?projection={projection}"/>
      </links>
      <name>SampleProvider</name>
      <localizedName>Sample Data Provider</localizedName>
      <localizedDescription>Example of a data provider</localizedDescription>
    </dataProvider>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/Act
ivityGraphProvider"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/ActivityGraphProvider?projection={projection}"/>
      </links>
      <name>ActivityGraphProvider</name>
      <localizedName>Activity Graph</localizedName>
      <localizedDescription>Activity Graph Adapter</localizedDescription>
    </dataProvider>
    <dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
```

```
?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider?projection={projection}"/>
      </links>
      <name>oracle.PropertiesServiceProvider</name>
      <localizedName>Property Service Data Provider</localizedName>
      <localizedDescription>Data provider implementation to integrate the
properties service for use in scenarios. </localizedDescription>
    </dataProvider>
  </items>
</dataProviders>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projectio
n=summary
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=ylp2M27pSyLJPv2X4cn9RYy7NRyftzz2vs7VCyJ1NQKJl4XQnjXd!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:10 GMT

{
  "dataProviders" : [ {
    "name" : "CMISProvider",
    "localizedName" : "Universal Content Provider (CMIS)",
    "localizedDescription" : "Provides access to content over CMIS connections.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/CMISProv
ider?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/CMISProv
```

```
ider",
      "rel" : "self"
    } ]
  }, {
    "name" : "SampleProvider",
    "localizedName" : "Sample Data Provider",
    "localizedDescription" : "Example of a data provider",
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/SamplePr
ovider?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/SamplePr
ovider",
      "rel" : "self"
    } ]
  }, {
    "name" : "ActivityGraphProvider",
    "localizedName" : "Activity Graph",
    "localizedDescription" : "Activity Graph Adapter",
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/Activity
GraphProvider?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/Activity
GraphProvider",
      "rel" : "self"
    } ]
  }, {
    "name" : "oracle.PropertiesServiceProvider",
    "localizedName" : "Property Service Data Provider",
    "localizedDescription" : "Data provider implementation to integrate the
properties service for use in scenarios. ",
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "links" : [ {
```

```
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider",
      "rel" : "self"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on=summary&startIndex=0&itemsPerPage=10",
    "rel" : "self"
  } ]
}
```

68.4.1.2 Getting a Single Data Provider's Metadata (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider?projection=summary
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 5792
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=L3b2M25L67PvznfyQNqHcyPh2J11rP51MMYlL1h80GvWsPxlHPvY!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 22:58:48 GMT

<?xml version="1.0" encoding="UTF-8"?>
<dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
  <links>
```

```
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider?projection=summary"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:providerConnections"
      resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:providerConnection"
      resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:connectionResources"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources?projection={projecti
on}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:connectionResource"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}?proj
ection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:resourceMethods"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}/meth
ods?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:resourceMethod"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}/meth
ods/{method-name}?projection={projection}"/>
  </links>
  <name>oracle.PropertiesServiceProvider</name>
  <localizedName>Property Service Data Provider</localizedName>
  <localizedDescription>Data provider implementation to integrate the properties
service for use in scenarios. </localizedDescription>
  <providerConnections
resourceType="urn:oracle:wcps:conductor:provider:providerConnections">
    <links>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
```

```
cle.PropertiesServiceProvider"
        rel="urn:oracle:webcenter:parent"
        resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider?projection={projection}"/>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections?projection={projection}"/>
    </links>
    <items>
      <providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/metadata"
            rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
            resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/metadata?proje
ction={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection?projection={pr
ojection}"/>
        </links>
        <name>LocalServerConnection</name>
      </providerConnection>
    </items>
  </providerConnections>
</dataProvider>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider?projection=details
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=R1P2M27Qd09Wh86s5rGdDFfBcbKtftPKvpdMsN2gcpLRN7XPyDnQ!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:12 GMT

{
  "name" : "oracle.PropertiesServiceProvider",
  "localizedName" : "Property Service Data Provider",
  "localizedDescription" : "Data provider implementation to integrate the
properties service for use in scenarios. ",
  "providerConnections" : [ {
    "name" : "LocalServerConnection",
    "connectionResources" : [ {
      "name" : "GetPropertiesResource",
      "resourceMethods" : {
        "resourceMethods" : [ {
          "name" : "GetProperty",
          "uniqueName" : "GetProperty(namespace,definition,set,property)",
          "returnType" : "java.io.Serializable",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          }, {
            "type" : "java.lang.String",
            "name" : "property",
            "localizedDescription" : "The property name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property exists\tnull will be returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
```

```
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)?projection={projec
tion}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetProperty",
          "uniqueName" : "GetProperty(definition,set,property)",
          "returnType" : "java.io.Serializable",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          }, {
            "type" : "java.lang.String",
            "name" : "property",
            "localizedDescription" : "The property name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property exists\tnull will be returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)",
            "rel" : "self"
          } ]
```

```
        }, {
          "name" : "execute",
          "uniqueName" : "execute()",
          "returnType" : "java.lang.Object",
          "parameters" : [ ],
          "localizedDescription" : "This is the default method executed if no
method is specified, which currently returns nothing.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySet",
          "uniqueName" : "GetPropertySet(definition,set)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name. If no
namespace, property set definition, or property set exists, null will be
returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
```

```
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySet",
          "uniqueName" : "GetPropertySet(namespace,definition,set)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name. If no
namespace, property set definition, or property set exists, null will be
returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)?projection={projection}"
,
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySets",
          "uniqueName" : "GetPropertySets(definition)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          } ],
          "localizedDescription" : "Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the request.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySets",
          "uniqueName" : "GetPropertySets(namespace,definition)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          } ],
          "localizedDescription" : "Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
```

```
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the request.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)",
            "rel" : "self"
          } ]
        } ],
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "links" : [ {
          "resourceType" :
"urn:oracle:wcps:conductor:provider:connectionResource",
          "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
          "capabilities" : "urn:oracle:restframework:read",
          "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
          "rel" : "urn:oracle:webcenter:parent"
        }, {
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
          "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
          "capabilities" : "urn:oracle:restframework:read",
          "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
          "rel" : "self"
        } ]
      },
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
```

```
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources?projection={p
rojection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
        "rel" : "self"
      } ]
    } ],
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionMetadata",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata?projection={pr
ojection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata",
      "rel" : "urn:oracle:wcps:conductor:provider:connectionMetadata"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection?projection={projection}
",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection",
      "rel" : "self"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
  "links" : [ {
```

```
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider?projection=details",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:providerConnections"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:providerConnection"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:connectionResources"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}?projection={p
rojection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:connectionResource"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}/methods?proje
ction={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:resourceMethods"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}/methods/{meth
```

```
od-name}?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:resourceMethod"
  } ]
}
```

68.4.1.3 Getting Information About a Data Provider Connection (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection?projection=summary
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 3896
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=QGswM25H1GLC61Qvbt08HLfJhfJdN50TSV8lRzWhpvbJG0vFJtSN!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 22:58:47 GMT

<?xml version="1.0" encoding="UTF-8"?>
<connection resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/metadata"
      rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
      resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/metadata?proje
ction={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection?projection=summary
"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection?projection={pr
ojection}"/>
  </links>
```

```
  <name>LocalServerConnection</name>
  <connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
    <links>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection"
        rel="urn:oracle:webcenter:parent"
        resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection?projection={pr
ojection}"/>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources?proj
ection={projection}"/>
    </links>
    <items>
      <connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources?proj
ection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource?projection={projection}"/>
        </links>
        <name>GetPropertiesResource</name>
      </connectionResource>
    </items>
  </connectionResources>
</connection>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection?projection=summary
accept-language: en
content-type: application/json
```

```
accept: application/json

Response:
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=tLqRM27fHTGT9nxHfW1TyHJv3522rhw84Z5RRcymb1dy3pLMFgp6!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:11 GMT

{
  "name" : "LocalServerConnection",
  "connectionResources" : [ {
    "name" : "GetPropertiesResource",
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources?projection={p
rojection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
      "rel" : "self"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionMetadata",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata?projection={pr
ojection}",
    "capabilities" : "urn:oracle:restframework:read",
```

```
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata",
    "rel" : "urn:oracle:wcps:conductor:provider:connectionMetadata"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection?projection={projection}
",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection?projection=summary",
    "rel" : "self"
  } ]
}
```

68.4.1.4 Getting Information about an Executable Resource for a Data Provider and Connection (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection/resources/GetPropertiesR
esource?projection=summary
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 11964
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=6NMkM25Ljb1yWPTMTSyMYksJjhtVKR90BhTv5MvGVjDHCVGyHGQG!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 22:58:48 GMT

<?xml version="1.0" encoding="UTF-8"?>
<connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources?proj
ection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
```

```
rtiesResource?projection=summary"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource?projection={projection}"/>
  </links>
  <name>GetPropertiesResource</name>
  <resourceMethods
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods">
    <links>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource"
        rel="urn:oracle:webcenter:parent"
        resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource?projection={projection}"/>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
    </links>
    <items>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetProperty(namespace,definition,set,property)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetProperty(namespace,definition,set,property)?projectio
n={projection}"/>
        </links>
        <name>GetProperty</name>
        <uniqueName>GetProperty(namespace,definition,set,property)</uniqueName>
```

```
        <parameters/>
      </resourceMethod>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetProperty(definition,set,property)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetProperty(definition,set,property)?projection={project
ion}"/>
        </links>
        <name>GetProperty</name>
        <uniqueName>GetProperty(definition,set,property)</uniqueName>
        <parameters/>
      </resourceMethod>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/execute()"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/execute()?projection={projection}"/>
        </links>
        <name>execute</name>
        <uniqueName>execute()</uniqueName>
        <parameters/>
      </resourceMethod>
      <resourceMethod
```

```
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySet(definition,set)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySet(definition,set)?projection={projection}"/
>
        </links>
        <name>GetPropertySet</name>
        <uniqueName>GetPropertySet(definition,set)</uniqueName>
        <parameters/>
      </resourceMethod>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySet(namespace,definition,set)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySet(namespace,definition,set)?projection={pro
jection}"/>
        </links>
        <name>GetPropertySet</name>
        <uniqueName>GetPropertySet(namespace,definition,set)</uniqueName>
        <parameters/>
      </resourceMethod>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
```

```
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySets(definition)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySets(definition)?projection={projection}"/>
        </links>
        <name>GetPropertySets</name>
        <uniqueName>GetPropertySets(definition)</uniqueName>
        <parameters/>
      </resourceMethod>
      <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySets(namespace,definition)"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySets(namespace,definition)?projection={projec
tion}"/>
        </links>
        <name>GetPropertySets</name>
        <uniqueName>GetPropertySets(namespace,definition)</uniqueName>
        <parameters/>
      </resourceMethod>
    </items>
  </resourceMethods>
</connectionResource>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection/resources/GetPropertiesR
esource?projection=summary
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=lP2dM27QwpNKslGK8xd818TLMmBhpdBzcLnZyncLXWLv6BrNjkq8!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:12 GMT

{
  "name" : "GetPropertiesResource",
  "resourceMethods" : {
    "resourceMethods" : [ {
      "name" : "GetProperty",
      "uniqueName" : "GetProperty(namespace,definition,set,property)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)?projection={projec
tion}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)",
        "rel" : "self"
      } ]
    }, {
      "name" : "GetProperty",
      "uniqueName" : "GetProperty(definition,set,property)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)",
        "rel" : "self"
      } ]
    }, {
      "name" : "execute",
      "uniqueName" : "execute()",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()",
        "rel" : "self"
      } ]
    }, {
      "name" : "GetPropertySet",
      "uniqueName" : "GetPropertySet(definition,set)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)",
        "rel" : "self"
      } ]
    }, {
      "name" : "GetPropertySet",
      "uniqueName" : "GetPropertySet(namespace,definition,set)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)?projection={projection}"
,
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)",
        "rel" : "self"
      } ]
    }, {
      "name" : "GetPropertySets",
      "uniqueName" : "GetPropertySets(definition)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
```

```
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)",
        "rel" : "self"
      } ]
    }, {
      "name" : "GetPropertySets",
      "uniqueName" : "GetPropertySets(namespace,definition)",
      "parameters" : [ ],
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)",
        "rel" : "self"
      } ]
    } ],
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
```

```
Resource?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
      "rel" : "self"
    } ]
  },
  "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources?projection={p
rojection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection=summary",
    "rel" : "self"
  } ]
}
```

68.4.1.5 Getting Method Information for an Executable Resource for a Data Provider and Connection (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection/resources/GetPropertiesR
esource/methods/GetPropertySet(namespace,definition,set)?projection=details
accept-language: en
```

```
content-type: application/xml
accept: application/xml

Response:
Status Code:200
Content-Length: 2243
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=rnBkM27d8dHMpZT4WdT0CLGythB2hRhQ2yjTmg6rh7BDfB1JTQjL!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 23:07:09 GMT

<?xml version="1.0" encoding="UTF-8"?>
<resourceMethod resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySet(namespace,definition,set)?projection=details"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySet(namespace,definition,set)?projection={pro
jection}"/>
  </links>
  <name>GetPropertySet</name>
  <uniqueName>GetPropertySet(namespace,definition,set)</uniqueName>
  <localizedDescription>Retrieves a property set for the current (or specified)
namespace, property set definition name, and property set name. If no namespace,
property set definition, or property set exists, null will be
returned.</localizedDescription>
  <parameters>
    <parameter>
      <localizedDescription>The namespace in which the property set definition and
property set exists.</localizedDescription>
      <name>namespace</name>
      <type>java.lang.String</type>
    </parameter>
    <parameter>
      <localizedDescription>The property set definition
name.</localizedDescription>
      <name>definition</name>
      <type>java.lang.String</type>
    </parameter>
    <parameter>
      <localizedDescription>The property set name.</localizedDescription>
      <name>set</name>
```

```
      <type>java.lang.String</type>
    </parameter>
  </parameters>
  <returnType>java.util.Map</returnType>
</resourceMethod>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider/connections/LocalServerConnection/resources/GetPropertiesR
esource/methods/GetPropertySet(namespace,definition,set)?projection=details
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=MD62M27BQLv9jxFR8K1SSXXyL70DwLGd7gpHvMh21xGM9gfppLR2!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:13 GMT

{
  "name" : "GetPropertySet",
  "uniqueName" : "GetPropertySet(namespace,definition,set)",
  "returnType" : "java.util.Map",
  "parameters" : [ {
    "type" : "java.lang.String",
    "name" : "namespace",
    "localizedDescription" : "The namespace in which the property set definition
and property set exists."
  }, {
    "type" : "java.lang.String",
    "name" : "definition",
    "localizedDescription" : "The property set definition name."
  }, {
    "type" : "java.lang.String",
    "name" : "set",
    "localizedDescription" : "The property set name."
  } ],
  "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name. If no
namespace, property set definition, or property set exists, null will be
returned.",
  "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
```

```
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)?projection={projection}"
,
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)?projection=details",
    "rel" : "self"
  } ]
}
```

68.4.1.6 Getting Entire Data Provider Information with Details Projection (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider?projection=details
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=T5nQM27dHT1yGdzP7yMy25l0QGKDt95kXymntN83tfhT72n1gkcQ!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/xml
Date: Wed, 13 Oct 2010 23:07:09 GMT

<?xml version="1.0" encoding="UTF-8"?>
<dataProvider resourceType="urn:oracle:wcps:conductor:provider:dataProvider">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider?projection=details"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:providerConnections"
```

```
      resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:providerConnection"
      resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:connectionResources"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources?projection={projecti
on}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:connectionResource"
      resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}?proj
ection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:resourceMethods"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}/meth
ods?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"
      rel="urn:oracle:wcps:conductor:provider:resourceMethod"
      resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProv
iders/{provider-name}/connections/{connection-name}/resources/{resource-name}/meth
ods/{method-name}?projection={projection}"/>
  </links>
  <name>oracle.PropertiesServiceProvider</name>
  <localizedName>Property Service Data Provider</localizedName>
  <localizedDescription>Data provider implementation to integrate the properties
service for use in scenarios. </localizedDescription>
  <providerConnections
resourceType="urn:oracle:wcps:conductor:provider:providerConnections">
    <links>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider"
        rel="urn:oracle:webcenter:parent"
        resourceType="urn:oracle:wcps:conductor:provider:dataProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider?projection={projection}"/>
      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections"
        rel="self"
        resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections?projection={projection}"/>
    </links>
    <items>
      <providerConnection
resourceType="urn:oracle:wcps:conductor:provider:providerConnection">
```

```
        <links>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections"
            rel="urn:oracle:webcenter:parent"
            resourceType="urn:oracle:wcps:conductor:provider:providerConnections"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections?projection={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/metadata"
            rel="urn:oracle:wcps:conductor:provider:connectionMetadata"
            resourceType="urn:oracle:wcps:conductor:provider:connectionMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/metadata?proje
ction={projection}"/>
          <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection"
            rel="self"
            resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection?projection={pr
ojection}"/>
        </links>
        <name>LocalServerConnection</name>
        <connectionResources
resourceType="urn:oracle:wcps:conductor:provider:connectionResources">
          <links>
            <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection"
              rel="urn:oracle:webcenter:parent"
              resourceType="urn:oracle:wcps:conductor:provider:providerConnection"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection?projection={pr
ojection}"/>
            <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
              rel="self"

resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources?proj
ection={projection}"/>
          </links>
          <items>
            <connectionResource
resourceType="urn:oracle:wcps:conductor:provider:connectionResource">
              <links>
                <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources"
```

```
                rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:connectionResources"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources?proj
ection={projection}"/>
              <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource"
                rel="self"

resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource?projection={projection}"/>
            </links>
            <name>GetPropertiesResource</name>
            <resourceMethods
resourceType="urn:oracle:wcps:conductor:provider:resourceMethods">
              <links>
                <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource"
                  rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:connectionResource"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource?projection={projection}"/>
                <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                  rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
              </links>
              <items>
                <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                  <links>
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                      rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
```

```
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetProperty(namespace,definition,set,property)"
                      rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetProperty(namespace,definition,set,property)?projectio
n={projection}"/>
                  </links>
                  <name>GetProperty</name>

<uniqueName>GetProperty(namespace,definition,set,property)</uniqueName>
                  <localizedDescription>Retrieves a property set for the current
(or specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property existsnull will be returned.</localizedDescription>
                  <parameters>
                    <parameter>
                      <localizedDescription>The namespace in which the property
set definition and property set exists.</localizedDescription>
                      <name>namespace</name>
                      <type>java.lang.String</type>
                    </parameter>
                    <parameter>
                      <localizedDescription>The property set definition
name.</localizedDescription>
                      <name>definition</name>
                      <type>java.lang.String</type>
                    </parameter>
                    <parameter>
                      <localizedDescription>The property set
name.</localizedDescription>
                      <name>set</name>
                      <type>java.lang.String</type>
                    </parameter>
                    <parameter>
                      <localizedDescription>The property
name.</localizedDescription>
                      <name>property</name>
                      <type>java.lang.String</type>
                    </parameter>
                  </parameters>
                  <returnType>java.io.Serializable</returnType>
                </resourceMethod>
                <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                  <links>
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                      rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
```

```
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetProperty(definition,set,property)"
                        rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetProperty(definition,set,property)?projection={project
ion}"/>
                    </links>
                    <name>GetProperty</name>
                    <uniqueName>GetProperty(definition,set,property)</uniqueName>
                    <localizedDescription>Retrieves a property set for the current
(or specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property existsnull will be returned.</localizedDescription>
                    <parameters>
                      <parameter>
                        <localizedDescription>The property set definition
name.</localizedDescription>
                        <name>definition</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property set
name.</localizedDescription>
                        <name>set</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property
name.</localizedDescription>
                        <name>property</name>
                        <type>java.lang.String</type>
                      </parameter>
                    </parameters>
                    <returnType>java.io.Serializable</returnType>
                  </resourceMethod>
                  <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                    <links>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                        rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
```

```
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/execute()"
                        rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/execute()?projection={projection}"/>
                    </links>
                    <name>execute</name>
                    <uniqueName>execute()</uniqueName>
                    <localizedDescription>This is the default method executed if
no method is specified, which currently returns nothing.</localizedDescription>
                    <parameters/>
                    <returnType>java.lang.Object</returnType>
                  </resourceMethod>
                  <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                    <links>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                        rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySet(definition,set)"
                        rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySet(definition,set)?projection={projection}"/
>
                    </links>
                    <name>GetPropertySet</name>
                    <uniqueName>GetPropertySet(definition,set)</uniqueName>
                    <localizedDescription>Retrieves a property set for the current
(or specified) namespace, property set definition name, and property set name. If
no namespace, property set definition, or property set exists, null will be
returned.</localizedDescription>
                    <parameters>
                      <parameter>
                        <localizedDescription>The property set definition
name.</localizedDescription>
                        <name>definition</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property set
name.</localizedDescription>
                        <name>set</name>
```

```
                        <type>java.lang.String</type>
                      </parameter>
                    </parameters>
                    <returnType>java.util.Map</returnType>
                  </resourceMethod>
                  <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                    <links>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                        rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                      <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySet(namespace,definition,set)"
                        rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySet(namespace,definition,set)?projection={pro
jection}"/>
                    </links>
                    <name>GetPropertySet</name>

<uniqueName>GetPropertySet(namespace,definition,set)</uniqueName>
                    <localizedDescription>Retrieves a property set for the current
(or specified) namespace, property set definition name, and property set name. If
no namespace, property set definition, or property set exists, null will be
returned.</localizedDescription>
                    <parameters>
                      <parameter>
                        <localizedDescription>The namespace in which the property
set definition and property set exists.</localizedDescription>
                        <name>namespace</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property set definition
name.</localizedDescription>
                        <name>definition</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property set
name.</localizedDescription>
                        <name>set</name>
                        <type>java.lang.String</type>
                      </parameter>
                    </parameters>
                    <returnType>java.util.Map</returnType>
```

```
                </resourceMethod>
                <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                  <links>
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                          rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySets(definition)"
                          rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySets(definition)?projection={projection}"/>
                  </links>
                  <name>GetPropertySets</name>
                  <uniqueName>GetPropertySets(definition)</uniqueName>
                  <localizedDescription>Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the
request.</localizedDescription>
                  <parameters>
                    <parameter>
                      <localizedDescription>The property set definition
name.</localizedDescription>
                      <name>definition</name>
                      <type>java.lang.String</type>
                    </parameter>
                  </parameters>
                  <returnType>java.util.Map</returnType>
                </resourceMethod>
                <resourceMethod
resourceType="urn:oracle:wcps:conductor:provider:resourceMethod">
                  <links>
                    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods"
                          rel="urn:oracle:webcenter:parent"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethods"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods?projection={projection}"/>
                    <link capabilities="urn:oracle:restframework:read"
```

```
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/ora
cle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetPrope
rtiesResource/methods/GetPropertySets(namespace,definition)"
                        rel="self"

resourceType="urn:oracle:wcps:conductor:provider:resourceMethod"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
/oracle.PropertiesServiceProvider/connections/LocalServerConnection/resources/GetP
ropertiesResource/methods/GetPropertySets(namespace,definition)?projection={projec
tion}"/>
                    </links>
                    <name>GetPropertySets</name>
                    <uniqueName>GetPropertySets(namespace,definition)</uniqueName>
                    <localizedDescription>Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the
request.</localizedDescription>
                    <parameters>
                      <parameter>
                        <localizedDescription>The namespace in which the property
set definition and property set exists.</localizedDescription>
                        <name>namespace</name>
                        <type>java.lang.String</type>
                      </parameter>
                      <parameter>
                        <localizedDescription>The property set definition
name.</localizedDescription>
                        <name>definition</name>
                        <type>java.lang.String</type>
                      </parameter>
                    </parameters>
                    <returnType>java.util.Map</returnType>
                  </resourceMethod>
                </items>
              </resourceMethods>
            </connectionResource>
          </items>
        </connectionResources>
      </providerConnection>
    </items>
  </providerConnections>
</dataProvider>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.Pr
opertiesServiceProvider?projection=details
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=GvbXM27BZxQhPJQqnv9Qmsn1j2G1Lvzhc67hpLp4gnjhjsypnTTm!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
```

```
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:07:13 GMT

{
  "name" : "oracle.PropertiesServiceProvider",
  "localizedName" : "Property Service Data Provider",
  "localizedDescription" : "Data provider implementation to integrate the
properties service for use in scenarios. ",
  "providerConnections" : [ {
    "name" : "LocalServerConnection",
    "connectionResources" : [ {
      "name" : "GetPropertiesResource",
      "resourceMethods" : {
        "resourceMethods" : [ {
          "name" : "GetProperty",
          "uniqueName" : "GetProperty(namespace,definition,set,property)",
          "returnType" : "java.io.Serializable",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          }, {
            "type" : "java.lang.String",
            "name" : "property",
            "localizedDescription" : "The property name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property exists\tnull will be returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)?projection={projec
tion}",
            "capabilities" : "urn:oracle:restframework:read",
```

```
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(namespace,definition,set,property)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetProperty",
          "uniqueName" : "GetProperty(definition,set,property)",
          "returnType" : "java.io.Serializable",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          }, {
            "type" : "java.lang.String",
            "name" : "property",
            "localizedDescription" : "The property name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name, and
property name.  If no namespace, property set definition, property set and
property exists\tnull will be returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetProperty(definition,set,property)",
            "rel" : "self"
          } ]
        }, {
          "name" : "execute",
          "uniqueName" : "execute()",
          "returnType" : "java.lang.Object",
          "parameters" : [ ],
          "localizedDescription" : "This is the default method executed if no
method is specified, which currently returns nothing.",
```

```
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/execute()",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySet",
          "uniqueName" : "GetPropertySet(definition,set)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name. If no
namespace, property set definition, or property set exists, null will be
returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
```

```
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(definition,set)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySet",
          "uniqueName" : "GetPropertySet(namespace,definition,set)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          }, {
            "type" : "java.lang.String",
            "name" : "set",
            "localizedDescription" : "The property set name."
          } ],
          "localizedDescription" : "Retrieves a property set for the current (or
specified) namespace, property set definition name, and property set name. If no
namespace, property set definition, or property set exists, null will be
returned.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)?projection={projection}"
,
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySet(namespace,definition,set)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySets",
```

```
          "uniqueName" : "GetPropertySets(definition)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          } ],
          "localizedDescription" : "Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the request.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(definition)",
            "rel" : "self"
          } ]
        }, {
          "name" : "GetPropertySets",
          "uniqueName" : "GetPropertySets(namespace,definition)",
          "returnType" : "java.util.Map",
          "parameters" : [ {
            "type" : "java.lang.String",
            "name" : "namespace",
            "localizedDescription" : "The namespace in which the property set
definition and property set exists."
          }, {
            "type" : "java.lang.String",
            "name" : "definition",
            "localizedDescription" : "The property set definition name."
          } ],
          "localizedDescription" : "Retrieves a Map of property sets for the
current (or specified) namespace and property set definition name that is keyed by
property set name. The size and start index of this list will be within the
boundaries of the specified start index and page size specified in the request.",
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
          "links" : [ {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
```

```
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
            "rel" : "urn:oracle:webcenter:parent"
          }, {
            "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
            "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)?projection={projection}",
            "capabilities" : "urn:oracle:restframework:read",
            "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods/GetPropertySets(namespace,definition)",
            "rel" : "self"
          } ]
        } ],
        "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
        "links" : [ {
          "resourceType" :
"urn:oracle:wcps:conductor:provider:connectionResource",
          "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
          "capabilities" : "urn:oracle:restframework:read",
          "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
          "rel" : "urn:oracle:webcenter:parent"
        }, {
          "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
          "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods?projection={projection}",
          "capabilities" : "urn:oracle:restframework:read",
          "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource/methods",
          "rel" : "self"
        } ]
      },
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
      "links" : [ {
        "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources?projection={p
rojection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
```

```
ropertiesServiceProvider/connections/LocalServerConnection/resources",
        "rel" : "urn:oracle:webcenter:parent"
      }, {
        "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
        "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource?projection={projection}",
        "capabilities" : "urn:oracle:restframework:read",
        "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/resources/GetProperties
Resource",
        "rel" : "self"
      } ]
    } ],
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:connectionMetadata",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata?projection={pr
ojection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection/metadata",
      "rel" : "urn:oracle:wcps:conductor:provider:connectionMetadata"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection?projection={projection}
",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider/connections/LocalServerConnection",
      "rel" : "self"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
```

```
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProvider",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders/oracle.P
ropertiesServiceProvider?projection=details",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnections",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:providerConnections"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:providerConnection",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:providerConnection"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResources",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:connectionResources"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:connectionResource",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}?projection={p
rojection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:connectionResource"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethods",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}/methods?proje
ction={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:resourceMethods"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:resourceMethod",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/{namespace}/dataProviders/{pr
ovider-name}/connections/{connection-name}/resources/{resource-name}/methods/{meth
od-name}?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "rel" : "urn:oracle:wcps:conductor:provider:resourceMethod"
  } ]
}
```

68.4.2 Using the Function Provider Management APIs

This section provides examples of using the Conductor's function provider APIs, and contains the following subsections:

- Section 68.4.2.1, "Getting a Collection of Function Providers (GET)"
- Section 68.4.2.2, "Getting Metadata for a Single Function Provider (GET)"

68.4.2.1 Getting a Collection of Function Providers (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proje
ction=summary
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=1NxqM2GHfPhT6VydXnQctrgCJ7FmxMLD3pPrRCgPrCLGRJGk9Rk5!2098846330;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/xml
Date: Wed, 13 Oct 2010 23:51:35 GMT

<?xml version="1.0" encoding="UTF-8"?>
<functionProviders
resourceType="urn:oracle:wcps:conductor:provider:functionProviders">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
?projection=summary"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
  </links>
  <items>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/agfunction:filterRecsByScore"
```

```
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/agfunction:filterRecsByScore?projection={projection}"/>
      </links>
      <category>agfunction</category>

<example>${agfunction:filterRecsByScore(recommendations,cutoffScore)}</example>
      <localizedDescription>Filter out Recommendations whose score is &gt;= the
input score</localizedDescription>
      <name>filterRecsByScore</name>
      <parameters/>

<returnType>oracle.wcps.activity.agrest.spy.jaxb.Recommendations</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/agfunction:getCMISLinksFromCommonItems"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/agfunction:getCMISLinksFromCommonItems?projection={projection}"/>
      </links>
      <category>agfunction</category>
      <example>${agfunction:getCMISLinksFromCommonItems(results)}</example>
      <localizedDescription>Return a List of URLs representing the common items as
CMIS objects</localizedDescription>
      <name>getCMISLinksFromCommonItems</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/agfunction:getCMISLinksFromRecommendations"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
```

```
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/agfunction:getCMISLinksFromRecommendations?projection={projection}"/>
      </links>
      <category>agfunction</category>

<example>${agfunction:getCMISLinksFromRecommendations(recommendations)}</example>
      <localizedDescription>Return a List of URLs representing the CMIS
objects</localizedDescription>
      <name>getCMISLinksFromRecommendations</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/agfunction:getContentIDs"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/agfunction:getContentIDs?projection={projection}"/>
      </links>
      <category>agfunction</category>
      <example>${agfunction:getContentIDs(agResults)}</example>
      <localizedDescription>Extract content IDs from input objects.  Includes
content of all types (WC.document, WC.wiki)</localizedDescription>
      <name>getContentIDs</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/agfunction:getContentIDsFiltered"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/agfunction:getContentIDsFiltered?projection={projection}"/>
      </links>
```

```
      <category>agfunction</category>

<example>${agfunction:getContentIDsFiltered(agResults,excludeClassURN)}</example>
      <localizedDescription>Extract content IDs from input
objects</localizedDescription>
      <name>getContentIDsFiltered</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:atomAsCMISObjects"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:atomAsCMISObjects?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:atomAsCMISObjects(atomFeed)}</example>
      <localizedDescription>ATOM feed as
List&lt;CMISObject&gt;</localizedDescription>
      <name>atomAsCMISObjects</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:atomAsEntries"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:atomAsEntries?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:atomAsEntries(atomFeed)}</example>
      <localizedDescription>ATOM feed as
```

```
List&lt;org.apache.abdera.model.Entry&gt;</localizedDescription>
      <name>atomAsEntries</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:atomAsFeed"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:atomAsFeed?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:atomAsFeed(atomFeed)}</example>
      <localizedDescription>ATOM feed as
org.apache.abdera.model.Feed</localizedDescription>
      <name>atomAsFeed</name>
      <parameters/>
      <returnType>org.apache.abdera.model.Feed</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:atomAsStreamUrls"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:atomAsStreamUrls?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:atomAsStreamUrls(atomFeed)}</example>
      <localizedDescription>ATOM feed as List&lt;String&gt; of
URLs</localizedDescription>
      <name>atomAsStreamUrls</name>
      <parameters/>
      <returnType>java.util.List</returnType>
```

```
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:getCMISQueryForDocIDs"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:getCMISQueryForDocIDs?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:getCMISQueryForDocIDs(repository,ids)}</example>
      <localizedDescription>Construct a CMIS query in the form of 'IN' query
syntax that will retrieve the documents for the array of input doc
IDs.</localizedDescription>
      <name>getCMISQueryForDocIDs</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/cmisfunction:getCMISQueryForDocIDsFromFullID"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/cmisfunction:getCMISQueryForDocIDsFromFullID?projection={projection}"/>
      </links>
      <category>cmisfunction</category>
      <example>${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}</example>
      <localizedDescription>Construct a CMIS query in the form of 'IN' query
syntax that will retrieve the documents for the array of input doc
IDs.</localizedDescription>
      <name>getCMISQueryForDocIDsFromFullID</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
```

```
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/collections:append"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/collections:append?projection={projection}"/>
      </links>
      <category>collections</category>
      <example>${collections:append(collection,item)}</example>
      <localizedDescription>Appends </localizedDescription>
      <name>append</name>
      <parameters/>
      <returnType>java.util.Collection</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/collections:new"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/collections:new?projection={projection}"/>
      </links>
      <category>collections</category>
      <example>${collections:new()}</example>
      <localizedDescription>Creates and returns a new
collection.</localizedDescription>
      <name>new</name>
      <parameters/>
      <returnType>java.util.Collection</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
```

```
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/collections:sort"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/collections:sort?projection={projection}"/>
      </links>
      <category>collections</category>
      <example>${collections:sort(list)}</example>
      <localizedDescription>Sorts the provided collection.</localizedDescription>
      <name>sort</name>
      <parameters/>
      <returnType>void</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/connectionsQA:getSampleConfig"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/connectionsQA:getSampleConfig?projection={projection}"/>
      </links>
      <category>connectionsQA</category>

<example>${connectionsQA:getSampleConfig(scenarioContext,connectionName)}</example
>
      <localizedDescription>Retrieves a sample configuration by
name.</localizedDescription>
      <name>getSampleConfig</name>
      <parameters/>

<returnType>oracle.wcps.test.providers.sample.SampleConnectionConfigImpl</returnTy
pe>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
```

```
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/connectionsQA:getSampleConfigs"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/connectionsQA:getSampleConfigs?projection={projection}"/>
      </links>
      <category>connectionsQA</category>
      <example>${connectionsQA:getSampleConfigs(scenarioContext)}</example>
      <localizedDescription>Retrieves all sample connection configurations for the
current namespace.</localizedDescription>
      <name>getSampleConfigs</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/connectionsQA:removeSampleConfig"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/connectionsQA:removeSampleConfig?projection={projection}"/>
      </links>
      <category>connectionsQA</category>

<example>${connectionsQA:removeSampleConfig(scenarioContext,connectionName)}</exam
ple>
      <localizedDescription>Removes a sample configuration with the specified name
in the current namespace.</localizedDescription>
      <name>removeSampleConfig</name>
      <parameters/>
      <returnType>void</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
```

```
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/connectionsQA:saveSampleConfig"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/connectionsQA:saveSampleConfig?projection={projection}"/>
      </links>
      <category>connectionsQA</category>

<example>${connectionsQA:saveSampleConfig(scenarioContext,connectionName,samplePro
pertyValue)}</example>
      <localizedDescription>Creates or saves a sample
configuration.</localizedDescription>
      <name>saveSampleConfig</name>
      <parameters/>

<returnType>oracle.wcps.test.providers.sample.SampleConnectionConfigImpl</returnTy
pe>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/magiceightball:response"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/magiceightball:response?projection={projection}"/>
      </links>
      <category>magiceightball</category>
      <example>${magiceightball:response(scenarioContext,responses)}</example>
      <localizedDescription>Provides clairvoyant responses to all your
questions</localizedDescription>
      <name>response</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
```

```
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/security:getUserRoles"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/security:getUserRoles?projection={projection}"/>
      </links>
      <category>security</category>
      <example>${security:getUserRoles()}</example>
      <localizedDescription>Returns an array of roles/groups that is part of the
current subject.</localizedDescription>
      <name>getUserRoles</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/security:isUserInRole"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/security:isUserInRole?projection={projection}"/>
      </links>
      <category>security</category>
      <example>${security:isUserInRole(role)}</example>
      <localizedDescription>Determines if the current user is a member in the
specified role or group name (role).</localizedDescription>
      <name>isUserInRole</name>
      <parameters/>
      <returnType>boolean</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"
```

```
href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/test:complexType"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/test:complexType?projection={projection}"/>
      </links>
      <category>test</category>
      <example>${test:complexType(arg0,arg1)}</example>
      <localizedDescription>Demonstrates how to return a complex
type.</localizedDescription>
      <name>complexType</name>
      <parameters/>
      <returnType>oracle.wcps.test.providers.functions.ComplexType</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/test:complexTypeList"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/test:complexTypeList?projection={projection}"/>
      </links>
      <category>test</category>
      <example>${test:complexTypeList(arg0)}</example>
      <localizedDescription>Demonstrates how to return a list of complex
types.</localizedDescription>
      <name>complexTypeList</name>
      <parameters/>
      <returnType>java.util.List</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/test:complexTypeMap"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
```

```
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/test:complexTypeMap?projection={projection}"/>
      </links>
      <category>test</category>
      <example>${test:complexTypeMap(arg0)}</example>
      <localizedDescription>Demonstrates how to return a map of complex
types.</localizedDescription>
      <name>complexTypeMap</name>
      <parameters/>
      <returnType>java.util.Map</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/test:concat"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/test:concat?projection={projection}"/>
      </links>
      <category>test</category>
      <example>${test:concat(arg0,arg1)}</example>
      <localizedDescription>Demonstrates how to return a complex
type.</localizedDescription>
      <name>concat</name>
      <parameters/>
      <returnType>java.lang.String</returnType>
    </functionProvider>
    <functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
      <links>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:webcenter:parent"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/test:sleep"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/test:sleep?projection={projection}"/>
      </links>
      <category>test</category>
```

```
      <example>${test:sleep(arg0)}</example>
      <localizedDescription>Pauses the current thread for the specified number of
milliseconds.</localizedDescription>
      <name>sleep</name>
      <parameters/>
      <returnType>void</returnType>
    </functionProvider>
  </items>
</functionProviders>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proje
ction=summary
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=Q9VFM2GLn2ny2CxbKvxKXYbYTMT4zvJ3QMgP2DzxH0GSkfct76cC!2098846330;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:51:36 GMT

{
  "functionProviders" : [ {
    "parameters" : [ ],
    "name" : "filterRecsByScore",
    "returnType" : "oracle.wcps.activity.agrest.spy.jaxb.Recommendations",
    "category" : "agfunction",
    "example" : "${agfunction:filterRecsByScore(recommendations,cutoffScore)}",
    "localizedDescription" : "Filter out Recommendations whose score is >= the
input score",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:filterRecsByScore?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:filterRecsByScore",
      "rel" : "self"
    } ]
  }, {
```

```
    "parameters" : [ ],
    "name" : "getCMISLinksFromCommonItems",
    "returnType" : "java.util.List",
    "category" : "agfunction",
    "example" : "${agfunction:getCMISLinksFromCommonItems(results)}",
    "localizedDescription" : "Return a List of URLs representing the common items
as CMIS objects",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getCMISLinksFromCommonItems?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getCMISLinksFromCommonItems",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getCMISLinksFromRecommendations",
    "returnType" : "java.util.List",
    "category" : "agfunction",
    "example" : "${agfunction:getCMISLinksFromRecommendations(recommendations)}",
    "localizedDescription" : "Return a List of URLs representing the CMIS
objects",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getCMISLinksFromRecommendations?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getCMISLinksFromRecommendations",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getContentIDs",
```

```
    "returnType" : "java.util.List",
    "category" : "agfunction",
    "example" : "${agfunction:getContentIDs(agResults)}",
    "localizedDescription" : "Extract content IDs from input objects.  Includes
content of all types (WC.document, WC.wiki)",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getContentIDs?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getContentIDs",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getContentIDsFiltered",
    "returnType" : "java.util.List",
    "category" : "agfunction",
    "example" : "${agfunction:getContentIDsFiltered(agResults,excludeClassURN)}",
    "localizedDescription" : "Extract content IDs from input objects",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getContentIDsFiltered?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/agfu
nction:getContentIDsFiltered",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "atomAsCMISObjects",
    "returnType" : "java.util.List",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:atomAsCMISObjects(atomFeed)}",
```

```
    "localizedDescription" : "ATOM feed as List<CMISObject>",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsCMISObjects?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsCMISObjects",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "atomAsEntries",
    "returnType" : "java.util.List",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:atomAsEntries(atomFeed)}",
    "localizedDescription" : "ATOM feed as  List<org.apache.abdera.model.Entry>",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsEntries?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsEntries",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "atomAsFeed",
    "returnType" : "org.apache.abdera.model.Feed",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:atomAsFeed(atomFeed)}",
    "localizedDescription" : "ATOM feed as  org.apache.abdera.model.Feed",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
```

```
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsFeed?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsFeed",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "atomAsStreamUrls",
    "returnType" : "java.util.List",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:atomAsStreamUrls(atomFeed)}",
    "localizedDescription" : "ATOM feed as List<String> of URLs",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsStreamUrls?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:atomAsStreamUrls",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getCMISQueryForDocIDs",
    "returnType" : "java.lang.String",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:getCMISQueryForDocIDs(repository,ids)}",
    "localizedDescription" : "Construct a CMIS query in the form of 'IN' query
syntax that will retrieve the documents for the array of input doc IDs.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
```

```
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:getCMISQueryForDocIDs?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:getCMISQueryForDocIDs",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getCMISQueryForDocIDsFromFullID",
    "returnType" : "java.lang.String",
    "category" : "cmisfunction",
    "example" : "${cmisfunction:getCMISQueryForDocIDsFromFullID(ids)}",
    "localizedDescription" : "Construct a CMIS query in the form of 'IN' query
syntax that will retrieve the documents for the array of input doc IDs.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:getCMISQueryForDocIDsFromFullID?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/cmis
function:getCMISQueryForDocIDsFromFullID",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "append",
    "returnType" : "java.util.Collection",
    "category" : "collections",
    "example" : "${collections:append(collection,item)}",
    "localizedDescription" : "Appends ",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
```

```
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:append?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:append",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "new",
    "returnType" : "java.util.Collection",
    "category" : "collections",
    "example" : "${collections:new()}",
    "localizedDescription" : "Creates and returns a new collection.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:new?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:new",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "sort",
    "returnType" : "void",
    "category" : "collections",
    "example" : "${collections:sort(list)}",
    "localizedDescription" : "Sorts the provided collection.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:sort?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:sort",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getSampleConfig",
    "returnType" : "oracle.wcps.test.providers.sample.SampleConnectionConfigImpl",
    "category" : "connectionsQA",
    "example" :
"${connectionsQA:getSampleConfig(scenarioContext,connectionName)}",
    "localizedDescription" : "Retrieves a sample configuration by name.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:getSampleConfig?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:getSampleConfig",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getSampleConfigs",
    "returnType" : "java.util.List",
    "category" : "connectionsQA",
    "example" : "${connectionsQA:getSampleConfigs(scenarioContext)}",
    "localizedDescription" : "Retrieves all sample connection configurations for
the current namespace.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:getSampleConfigs?projection={projection}",
```

```
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:getSampleConfigs",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "removeSampleConfig",
    "returnType" : "void",
    "category" : "connectionsQA",
    "example" :
"${connectionsQA:removeSampleConfig(scenarioContext,connectionName)}",
    "localizedDescription" : "Removes a sample configuration with the specified
name in the current namespace.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:removeSampleConfig?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:removeSampleConfig",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "saveSampleConfig",
    "returnType" : "oracle.wcps.test.providers.sample.SampleConnectionConfigImpl",
    "category" : "connectionsQA",
    "example" :
"${connectionsQA:saveSampleConfig(scenarioContext,connectionName,samplePropertyVal
ue)}",
    "localizedDescription" : "Creates or saves a sample configuration.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:saveSampleConfig?projection={projection}",
```

```
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/conn
ectionsQA:saveSampleConfig",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "response",
    "returnType" : "java.lang.String",
    "category" : "magiceightball",
    "example" : "${magiceightball:response(scenarioContext,responses)}",
    "localizedDescription" : "Provides clairvoyant responses to all your
questions",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/magi
ceightball:response?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/magi
ceightball:response",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "getUserRoles",
    "returnType" : "java.util.List",
    "category" : "security",
    "example" : "${security:getUserRoles()}",
    "localizedDescription" : "Returns an array of roles/groups that is part of the
current subject.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/secu
rity:getUserRoles?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/secu
rity:getUserRoles",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "isUserInRole",
    "returnType" : "boolean",
    "category" : "security",
    "example" : "${security:isUserInRole(role)}",
    "localizedDescription" : "Determines if the current user is a member in the
specified role or group name (role).",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/secu
rity:isUserInRole?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/secu
rity:isUserInRole",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "complexType",
    "returnType" : "oracle.wcps.test.providers.functions.ComplexType",
    "category" : "test",
    "example" : "${test:complexType(arg0,arg1)}",
    "localizedDescription" : "Demonstrates how to return a complex type.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexType?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexType",
      "rel" : "self"
```

```
    } ]
  }, {
    "parameters" : [ ],
    "name" : "complexTypeList",
    "returnType" : "java.util.List",
    "category" : "test",
    "example" : "${test:complexTypeList(arg0)}",
    "localizedDescription" : "Demonstrates how to return a list of complex
types.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexTypeList?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexTypeList",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "complexTypeMap",
    "returnType" : "java.util.Map",
    "category" : "test",
    "example" : "${test:complexTypeMap(arg0)}",
    "localizedDescription" : "Demonstrates how to return a map of complex types.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexTypeMap?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:complexTypeMap",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
```

```
    "name" : "concat",
    "returnType" : "java.lang.String",
    "category" : "test",
    "example" : "${test:concat(arg0,arg1)}",
    "localizedDescription" : "Demonstrates how to return a complex type.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:concat?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:concat",
      "rel" : "self"
    } ]
  }, {
    "parameters" : [ ],
    "name" : "sleep",
    "returnType" : "void",
    "category" : "test",
    "example" : "${test:sleep(arg0)}",
    "localizedDescription" : "Pauses the current thread for the specified number
of milliseconds.",
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:sleep?projection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/test
:sleep",
      "rel" : "self"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection=summary",
    "rel" : "self"
  } ]
}
```

68.4.2.2 Getting Metadata for a Single Function Provider (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/colle
ctions:append?projection=details
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 1433
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=xf4NM2GL2yrTY3QPCZ591Ht8h7MVTSW9GGzR6Tm12WtFLvD0vnng!2098846330;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 23:51:36 GMT

<?xml version="1.0" encoding="UTF-8"?>
<functionProvider
resourceType="urn:oracle:wcps:conductor:provider:functionProvider">
  <links>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
      rel="urn:oracle:webcenter:parent"
      resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
/collections:append?projection=details"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:provider:functionProvider"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders/collections:append?projection={projection}"/>
  </links>
  <category>collections</category>
  <example>${collections:append(collection,item)}</example>
  <localizedDescription>Appends </localizedDescription>
  <name>append</name>
  <parameters>
    <parameter>
```

```
      <localizedDescription>The collection to append to.</localizedDescription>
      <name>collection</name>
      <type>java.util.Collection</type>
    </parameter>
    <parameter>
      <localizedDescription>The item or object to append.</localizedDescription>
      <name>item</name>
      <type>java.lang.Object</type>
    </parameter>
  </parameters>
  <returnType>java.util.Collection</returnType>
</functionProvider>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/colle
ctions:append?projection=details
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=BCz3M2GJWQnS3jQV1DvMsJynJXGnlJ6MrmrST4qxm8L7NxbgJh9T!2098846330;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 23:51:37 GMT

{
  "parameters" : [ {
    "type" : "java.util.Collection",
    "name" : "collection",
    "localizedDescription" : "The collection to append to."
  }, {
    "type" : "java.lang.Object",
    "name" : "item",
    "localizedDescription" : "The item or object to append."
  } ],
  "name" : "append",
  "returnType" : "java.util.Collection",
  "category" : "collections",
  "example" : "${collections:append(collection,item)}",
  "localizedDescription" : "Appends ",
  "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProvider",
    "template" :
```

```
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:append?projection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders/coll
ections:append?projection=details",
    "rel" : "self"
  } ]
}
```

68.4.3 Using the Namespace Management APIs

This section provides examples of using the Conductor's namespace management APIs, and contains the following subsections:

- Section 68.4.3.1, "Getting a Collection of Namespaces (GET)"
- Section 68.4.3.2, "Retrieving a Single Namespace (GET)"
- Section 68.4.3.3, "Creating a Namespace (POST)"
- Section 68.4.3.4, "Deleting a Namespace (DELETE)"

68.4.3.1 Getting a Collection of Namespaces (GET)

XML

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=JVqNM0HGTxxrXJJsy2cLCPjRTqJzDQ6pFytMMNy3JQTph671rTC2!-1838429496;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/xml
Date: Tue, 12 Oct 2010 20:40:06 GMT

<?xml version="1.0" encoding="UTF-8"?>
<namespaces resourceType="urn:oracle:wcps:conductor:namespaces">
  <links>
    <link capabilities="urn:oracle:restframework:read"
      href="http://localhost:7001/wcps/api/conductor/resourceIndex"
      rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:resourceIndex"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"

href="http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&amp;itemsPe
rPage=10"
      rel="self" resourceType="urn:oracle:wcps:conductor:namespaces"
template="http://localhost:7001/wcps/api/conductor/namespaces?startIndex={startInd
ex}&amp;itemsPerPage={itemsPerPage}"/>
  </links>
```

```
  <items>
    <namespace resourceType="urn:oracle:wcps:conductor:namespace">
      <links>
        <link capabilities="urn:oracle:restframework:read"
          href="http://localhost:7001/wcps/api/conductor/namespaces"
          rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:namespaces"/>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:delete"
          href="http://localhost:7001/wcps/api/conductor/namespaces/JDEV"
          rel="self" resourceType="urn:oracle:wcps:conductor:namespace"/>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:create"

href="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/scenarios"
          rel="urn:oracle:wcps:conductor:scenarioMetadatas"
          resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
template="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/scenarios?tag={
tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={ite
msPerPage}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/dataProviders"
          rel="urn:oracle:wcps:conductor:provider:dataProviders"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/dataProviders?p
rojection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/functionProviders"
          rel="urn:oracle:wcps:conductor:provider:functionProviders"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/JDEV/functionProvide
rs?projection={projection}"/>
      </links>
      <name>JDEV</name>
    </namespace>
    <namespace resourceType="urn:oracle:wcps:conductor:namespace">
      <links>
        <link capabilities="urn:oracle:restframework:read"
          href="http://localhost:7001/wcps/api/conductor/namespaces"
          rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:namespaces"/>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:delete"
          href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle"
          rel="self" resourceType="urn:oracle:wcps:conductor:namespace"/>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:create"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios"
          rel="urn:oracle:wcps:conductor:scenarioMetadatas"
          resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag
={tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={i
temsPerPage}"/>
```

```
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
          rel="urn:oracle:wcps:conductor:provider:dataProviders"
          resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
          rel="urn:oracle:wcps:conductor:provider:functionProviders"
          resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
      </links>
      <name>Oracle</name>
    </namespace>
  </items>
</namespaces>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=3yvMM0HLChkXHS7R2VGn4F6lFKHwTGP2NwjC2Lf7J2jpvLTW9GvV!-1838429496;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Tue, 12 Oct 2010 20:40:11 GMT

{
  "namespaces" : [ {
    "name" : "JDEV",
    "resourceType" : "urn:oracle:wcps:conductor:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:namespaces",
      "capabilities" : "urn:oracle:restframework:read",
      "href" : "http://localhost:7001/wcps/api/conductor/namespaces",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
      "href" : "http://localhost:7001/wcps/api/conductor/namespaces/JDEV",
      "rel" : "self"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/scenarios?tag={tag}&proj
ection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
      "capabilities" : "urn:oracle:restframework:read
```

```
urn:oracle:restframework:create",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/scenarios",
      "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/dataProviders?projection
={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/dataProviders",
      "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/functionProviders?projec
tion={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/JDEV/functionProviders",
      "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
    } ]
  }, {
    "name" : "Oracle",
    "resourceType" : "urn:oracle:wcps:conductor:namespace",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:namespaces",
      "capabilities" : "urn:oracle:restframework:read",
      "href" : "http://localhost:7001/wcps/api/conductor/namespaces",
      "rel" : "urn:oracle:webcenter:parent"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:namespace",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
      "href" : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle",
      "rel" : "self"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&pr
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios",
      "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
      "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
```

```
ection={projection}",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
      "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
    } ]
  }],
  "resourceType" : "urn:oracle:wcps:conductor:namespaces",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:resourceIndex",
    "capabilities" : "urn:oracle:restframework:read",
    "href" : "http://localhost:7001/wcps/api/conductor/resourceIndex",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:namespaces",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces?startIndex={startIndex}&items
PerPage={itemsPerPage}",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10"
,
    "rel" : "self"
  } ]
}
```

68.4.3.2 Retrieving a Single Namespace (GET)

XML

Request:

```
GET http://localhost:7001/wcps/api/conductor/namespaces/Oracle
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=qzMqM0HMPYBlKQGMJvnQMS4rxXS4Jn1QnBjW850G1vnSjt4lHbLy!-1838429496;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Tue, 12 Oct 2010 20:40:12 GMT

{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:conductor:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:namespaces",
    "capabilities" : "urn:oracle:restframework:read",
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:namespace",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
```

```
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&pr
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios",
    "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
  } ]
}
```

JSON

Request:

```
GET http://localhost:7001/wcps/api/conductor/namespaces/Oracle
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=qzMqM0HMPYBlKQGMJvnQMS4rxXS4Jn1QnBjW850G1vnSjt4lHbLy!-1838429496;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Tue, 12 Oct 2010 20:40:12 GMT

{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:conductor:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:namespaces",
    "capabilities" : "urn:oracle:restframework:read",
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:namespace",
```

```
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&pr
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
    "href" :
"http:://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios",
    "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
  } ]
}
```

68.4.3.3 Creating a Namespace (POST)

XML

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:conductor:namespace">
  <name>Oracle</name>
</namespace>
```

Response:

```
Status Code:201
Content-Length: 1753
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=RW6QM0HF5x2KLvNSZJVncTpSXbl1rKY3tTfXyH94yY6Qv6ddGJkY!-1838429496;
path=/wcps; HttpOnly
Content-Type: application/xml
```

```
Location: http://localhost:7001/wcps/api/conductor/namespaces/Oracle
Date: Tue, 12 Oct 2010 20:40:05 GMT

<?xml version="1.0" encoding="UTF-8"?>
<namespace resourceType="urn:oracle:wcps:conductor:namespace">
  <links>
    <link capabilities="urn:oracle:restframework:read"
      href="http://localhost:7001/wcps/api/conductor/namespaces"
      rel="urn:oracle:webcenter:parent"
resourceType="urn:oracle:wcps:conductor:namespaces"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:delete"
      href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle"
      rel="self" resourceType="urn:oracle:wcps:conductor:namespace"/>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"
      href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios"
      rel="urn:oracle:wcps:conductor:scenarioMetadatas"
      resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag
={tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={i
temsPerPage}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders"
      rel="urn:oracle:wcps:conductor:provider:dataProviders"
      resourceType="urn:oracle:wcps:conductor:provider:dataProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders
?projection={projection}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders
"
      rel="urn:oracle:wcps:conductor:provider:functionProviders"
      resourceType="urn:oracle:wcps:conductor:provider:functionProviders"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProvi
ders?projection={projection}"/>
  </links>
  <name>Oracle</name>
</namespace>
```

JSON

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces?startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json

{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:conductor:namespace"
}
```

Response:

```
Status Code:201
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=xh18M0HKT86f4y1bWqZnBmzHXvNnvh0p7y9RLTmQfbRQC3fh79gx!-1838429496;
path=/wcps; HttpOnly
```

```
Transfer-Encoding: chunked
Content-Type: application/json
Location: http://localhost:7001/wcps/api/conductor/namespaces/Oracle
Date: Tue, 12 Oct 2010 20:40:10 GMT

{
  "name" : "Oracle",
  "resourceType" : "urn:oracle:wcps:conductor:namespace",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:namespaces",
    "capabilities" : "urn:oracle:restframework:read",
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces",
    "rel" : "urn:oracle:webcenter:parent"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:namespace",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:delete",
    "href" : "http://localhost:7001/wcps/api/conductor/namespaces/Oracle",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&pr
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios",
    "rel" : "urn:oracle:wcps:conductor:scenarioMetadatas"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:dataProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders?projecti
on={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/dataProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:dataProviders"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:provider:functionProviders",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders?proj
ection={projection}",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/functionProviders",
    "rel" : "urn:oracle:wcps:conductor:provider:functionProviders"
  } ]
}
```

68.4.3.4 Deleting a Namespace (DELETE)

XML/JSON

Request:

```
DELETE http://localhost:7001/wcps/api/conductor/namespaces/Oracle
accept-language: en
content-type: application/xml
accept: application/xml
```

```
Response:
Status Code:204
Content-Length: 0
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
Set-Cookie:
JSESSIONID=SMctM0HJxYxSvkMszjrfpl5zJ3Q0v25NhB7QPRTnJFLmd1tK1hYC!-1838429496;
path=/wcps; HttpOnly
Date: Tue, 12 Oct 2010 20:40:09 GMT
```

68.4.4 Using the Scenario Management APIs

This section provides examples of using the Conductor's scenario management APIs, and contains the following susbsections:

- Section 68.4.4.1, "Getting a Collection of Scenarios (GET)"
- Section 68.4.4.2, "Retrieving a Single Scenario (GET)"
- Section 68.4.4.3, "Creating a Scenario (POST)"
- Section 68.4.4.4, "Updating a Scenario (PUT)"
- Section 68.4.4.5, "Deleting a Scenario (DELETE)"
- Section 68.4.4.6, "Execute a Scenario (POST)"

68.4.4.1 Getting a Collection of Scenarios (GET)

XML

```
Request:
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=su
mmary&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml

Response:
Status Code:200
Content-Length: 1915
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=LJq1M1TGZp7fClyTJ2wggGczv2r8N0X1jsPYlg8kpGFWwGMvrQsM!21637120;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 15:43:02 GMT

<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadatas
  resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:create"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?project
ion=summary&amp;startIndex=0&amp;itemsPerPage=10"
```

```
      rel="self"
      resourceType="urn:oracle:wcps:conductor:scenarioMetadatas"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag
={tag}&amp;projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={i
temsPerPage}"/>
  </links>
  <items>
    <scenarioMetadata resourceType="urn:oracle:wcps:conductor:scenarioMetadata">
      <links>
        <link
          capabilities="urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World"
          rel="self"
          resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hel
lo
World?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={itemsP
erPage}"/>
        <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World/definition"
          rel="alternate" resourceType="urn:oracle:wcps:conductor:scenario"/>
      </links>
      <status>PUBLISHED</status>
      <createdAuthor>weblogic</createdAuthor>
      <createdDateTime>2010-10-13T09:43:01.617-06:00</createdDateTime>
      <updatedAuthor>weblogic</updatedAuthor>
      <updatedDateTime>2010-10-13T09:43:01.617-06:00</updatedDateTime>
      <scenario>
        <comments/>
        <name>Hello World</name>
        <tags/>
        <input-parameters/>
      </scenario>
    </scenarioMetadata>
  </items>
</scenarioMetadatas>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=su
mmary&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=0VBvM2ZMwfWC6bPnZ8QDWv7q4h7WhmTMPvb0Lyc9n8t0lnyTtGLy!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
```

```
Date: Wed, 13 Oct 2010 20:39:40 GMT

{
  "scenarioMetadatas" : [ {
    "scenario" : {
      "tags" : [ ],
      "inputParameters" : [ ],
      "name" : "Hello World",
      "comments" : "This scenario returns a Hello World string."
    },
    "status" : "PUBLISHED",
    "createdAuthor" : "weblogic",
    "createdDateTime" : "2010-10-13T14:39:40.103-0600",
    "updatedAuthor" : "weblogic",
    "updatedDateTime" : "2010-10-13T14:39:40.103-0600",
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
    "links" : [ {
      "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
      "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello
World?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"
,
      "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d",
      "rel" : "self"
    }, {
      "resourceType" : "urn:oracle:wcps:conductor:scenario",
      "capabilities" : "urn:oracle:restframework:read",
      "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d/definition",
      "rel" : "alternate"
    } ]
  } ],
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadatas",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?tag={tag}&pr
ojection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}",
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:create",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=s
ummary&startIndex=0&itemsPerPage=10",
    "rel" : "self"
  } ]
}
```

68.4.4.2 Retrieving a Single Scenario (GET)

XML

Request:

```
GET
```

```
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:200
Content-Length: 1725
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=mS7LM2hYbWDd8xvmKYGSGVLXJ2qYDTXNGLy3lqknyp8fYStlBQNT!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 20:45:44 GMT

<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadata
  resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete urn:oracle:restframework:execute"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World?projection=details&amp;startIndex=0&amp;itemsPerPage=10"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hel
lo
World?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={itemsP
erPage}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World/definition"
      rel="alternate" resourceType="urn:oracle:wcps:conductor:scenario"/>
  </links>
  <status>PUBLISHED</status>
  <createdAuthor>weblogic</createdAuthor>
  <createdDateTime>2010-10-13T14:45:42.525-06:00</createdDateTime>
  <updatedAuthor>weblogic</updatedAuthor>
  <updatedDateTime>2010-10-13T14:45:42.525-06:00</updatedDateTime>
  <scenario>
    <comments>This scenario returns a Hello World string.</comments>
    <body>
      <return>
        <comments>This is what is returned by the scenario.  A static string with
the input parameter embedded.</comments>
        <expression>Hello World with input '${input}'!</expression>
      </return>
    </body>
    <name>Hello World</name>
    <tags>
      <tag>hello</tag>
      <tag>world</tag>
    </tags>
    <input-parameters>
```

```
      <parameter required="true">input</parameter>
    </input-parameters>
  </scenario>
</scenarioMetadata>
```

JSON

Request:

```
GET
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=LJnKM2hppVQL9ph88JWCymz8M1TQqcQjSbD03zJQ9z6yyT314Zy4!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 20:45:50 GMT

{
  "scenario" : {
    "tags" : [ "hello", "world" ],
    "inputParameters" : [ {
      "value" : "input",
      "required" : true
    } ],
    "name" : "Hello World",
    "body" : {
      "statements" : [ {
        "concreteType" : "urn:oracle:wcps:conductor:scenario:schema:return",
        "expression" : "Hello World with input '${input}'!",
        "comments" : "This is what is returned by the scenario.  A static string
with the input parameter embedded."
      } ]
    },
    "comments" : "This scenario returns a Hello World string."
  },
  "status" : "PUBLISHED",
  "createdAuthor" : "weblogic",
  "createdDateTime" : "2010-10-13T14:45:48.681-0600",
  "updatedAuthor" : "weblogic",
  "updatedDateTime" : "2010-10-13T14:45:48.681-0600",
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello
World?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"
,
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d?projection=details&startIndex=0&itemsPerPage=10",
```

```
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenario",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d/definition",
    "rel" : "alternate"
  } ]
}
```

68.4.4.3 Creating a Scenario (POST)

XML

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=de
tails&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadata
  resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <scenario>
    <comments>This scenario returns a Hello World string.</comments>
    <body>
      <return>
        <comments>This is what is returned by the scenario.  A static string with
the input parameter embedded.</comments>
        <expression>Hello World with input '${input}'!</expression>
      </return>
    </body>
    <name>Hello World</name>
    <tags>
      <tag>hello</tag>
      <tag>world</tag>
    </tags>
    <input-parameters>
      <parameter required="true">input</parameter>
    </input-parameters>
  </scenario>
</scenarioMetadata>
```

Response:

```
Status Code:201
Content-Length: 1669
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=L41NM2hW6p8FtvJmjRxKkZNF2LJnnY4JhGJCT1xDG4prwf1T1g3D!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Location:
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
Date: Wed, 13 Oct 2010 20:45:42 GMT
```

```
<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadata
  resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete urn:oracle:restframework:execute"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hel
lo
World?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={itemsP
erPage}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World/definition"
      rel="alternate" resourceType="urn:oracle:wcps:conductor:scenario"/>
  </links>
  <status>PUBLISHED</status>
  <createdAuthor>weblogic</createdAuthor>
  <createdDateTime>2010-10-13T14:45:42.525-06:00</createdDateTime>
  <updatedAuthor>weblogic</updatedAuthor>
  <updatedDateTime>2010-10-13T14:45:42.525-06:00</updatedDateTime>
  <scenario>
    <comments>This scenario returns a Hello World string.</comments>
    <body>
      <return>
        <comments>This is what is returned by the scenario.  A static string with
the input parameter embedded.</comments>
        <expression>Hello World with input '${input}'!</expression>
      </return>
    </body>
    <name>Hello World</name>
    <tags>
      <tag>hello</tag>
      <tag>world</tag>
    </tags>
    <input-parameters>
      <parameter required="true">input</parameter>
    </input-parameters>
  </scenario>
</scenarioMetadata>
```

JSON

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios?projection=de
tails&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json

{
```

```
  "scenario" : {
    "tags" : [ "hello", "world" ],
    "inputParameters" : [ {
      "value" : "input",
      "required" : true
    } ],
    "name" : "Hello World",
    "body" : {
      "statements" : [ {
        "concreteType" : "urn:oracle:wcps:conductor:scenario:schema:return",
        "expression" : "Hello World with input '${input}'!",
        "comments" : "This is what is returned by the scenario.  A static string
with the input parameter embedded."
      } ]
    },
    "comments" : "This scenario returns a Hello World string."
  },
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata"
}
```

Response:

```
Status Code:201
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=XrT7M2hcNRzpmCKF7SxPNs9phBvfPlsnffy98nRLDL8G52wHwd1y!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Location:
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
Date: Wed, 13 Oct 2010 20:45:48 GMT

{
  "scenario" : {
    "tags" : [ "hello", "world" ],
    "inputParameters" : [ {
      "value" : "input",
      "required" : true
    } ],
    "name" : "Hello World",
    "body" : {
      "statements" : [ {
        "concreteType" : "urn:oracle:wcps:conductor:scenario:schema:return",
        "expression" : "Hello World with input '${input}'!",
        "comments" : "This is what is returned by the scenario.  A static string
with the input parameter embedded."
      } ]
    },
    "comments" : "This scenario returns a Hello World string."
  },
  "status" : "PUBLISHED",
  "createdAuthor" : "weblogic",
  "createdDateTime" : "2010-10-13T14:45:48.681-0600",
  "updatedAuthor" : "weblogic",
  "updatedDateTime" : "2010-10-13T14:45:48.681-0600",
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello
```

```
World?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"
,
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenario",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d/definition",
    "rel" : "alternate"
  } ]
}
```

68.4.4.4 Updating a Scenario (PUT)

XML

Request:

```
PUT
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadata
  resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <scenario>
    <comments>This scenario returns a Hello World string.</comments>
    <body>
      <assign-variable>
        <variable>initVariable</variable>
        <expression>Hello World with input '${input}'!</expression>
      </assign-variable>
      <return>
        <comments>This is what is returned by the scenario.  A static string with
the input parameter embedded.</comments>
        <expression>${initVariable}</expression>
      </return>
    </body>
    <name>Hello World</name>
    <tags>
      <tag>hello</tag>
      <tag>world</tag>
    </tags>
    <input-parameters>
      <parameter required="true">input</parameter>
    </input-parameters>
  </scenario>
</scenarioMetadata>
```

Response:

```
Status Code:200
Content-Length: 1777
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=CnGYM2hYBBJ2hNPGLkcDG1WWvhP41gBDvBsCwmKpdHvtCJtQL67L!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 20:45:44 GMT

<?xml version="1.0" encoding="UTF-8"?>
<scenarioMetadata
  resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
  xmlns:ns2="http://xmlns.oracle.com/wcps/conductor/common/1.0.0"
xmlns:ns3="http://xmlns.oracle.com/wcps/conductor/scenarios/1.0.0">
  <links>
    <link
      capabilities="urn:oracle:restframework:read urn:oracle:restframework:update
urn:oracle:restframework:delete urn:oracle:restframework:execute"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World"
      rel="self"
      resourceType="urn:oracle:wcps:conductor:scenarioMetadata"
template="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hel
lo
World?projection={projection}&amp;startIndex={startIndex}&amp;itemsPerPage={itemsP
erPage}"/>
    <link capabilities="urn:oracle:restframework:read"

href="http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%2
0World/definition"
      rel="alternate" resourceType="urn:oracle:wcps:conductor:scenario"/>
  </links>
  <status>PUBLISHED</status>
  <createdAuthor>weblogic</createdAuthor>
  <createdDateTime>2010-10-13T14:45:42.525-06:00</createdDateTime>
  <updatedAuthor>weblogic</updatedAuthor>
  <updatedDateTime>2010-10-13T14:45:44.471-06:00</updatedDateTime>
  <scenario>
    <comments>This scenario returns a Hello World string.</comments>
    <body>
      <assign-variable>
        <variable>initVariable</variable>
        <expression>Hello World with input '${input}'!</expression>
      </assign-variable>
      <return>
        <comments>This is what is returned by the scenario.  A static string with
the input parameter embedded.</comments>
        <expression>${initVariable}</expression>
      </return>
    </body>
    <name>Hello World</name>
    <tags>
      <tag>hello</tag>
      <tag>world</tag>
    </tags>
    <input-parameters>
      <parameter required="true">input</parameter>
```

```
    </input-parameters>
  </scenario>
</scenarioMetadata>
```

JSON

Request:

```
PUT
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json

{
  "scenario" : {
    "tags" : [ "hello", "world" ],
    "inputParameters" : [ {
      "value" : "input",
      "required" : true
    } ],
    "name" : "Hello World",
    "body" : {
      "statements" : [ {
        "concreteType" :
"urn:oracle:wcps:conductor:scenario:schema:variable-assignment",
        "expression" : "Hello World with input '${input}'!",
        "variable" : "initVariable"
      }, {
        "concreteType" : "urn:oracle:wcps:conductor:scenario:schema:return",
        "expression" : "${initVariable}",
        "comments" : "This is what is returned by the scenario.  A static string
with the input parameter embedded."
      } ]
    },
    "comments" : "This scenario returns a Hello World string."
  },
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata"
}
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=sJBGM2hpTwwM7l3Q7HNnQ8lDpjkqQvkJy4h1Wr11KJFx1gd2XDSY!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 20:45:50 GMT

{
  "scenario" : {
    "tags" : [ "hello", "world" ],
    "inputParameters" : [ {
      "value" : "input",
      "required" : true
    } ],
    "name" : "Hello World",
    "body" : {
      "statements" : [ {
        "concreteType" :
```

```
"urn:oracle:wcps:conductor:scenario:schema:variable-assignment",
        "expression" : "Hello World with input '${input}'!",
        "variable" : "initVariable"
      }, {
        "concreteType" : "urn:oracle:wcps:conductor:scenario:schema:return",
        "expression" : "${initVariable}",
        "comments" : "This is what is returned by the scenario.  A static string
with the input parameter embedded."
      } ]
    },
    "comments" : "This scenario returns a Hello World string."
  },
  "status" : "PUBLISHED",
  "createdAuthor" : "weblogic",
  "createdDateTime" : "2010-10-13T14:45:48.681-0600",
  "updatedAuthor" : "weblogic",
  "updatedDateTime" : "2010-10-13T14:45:50.533-0600",
  "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
  "links" : [ {
    "resourceType" : "urn:oracle:wcps:conductor:scenarioMetadata",
    "template" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello
World?projection={projection}&startIndex={startIndex}&itemsPerPage={itemsPerPage}"
,
    "capabilities" : "urn:oracle:restframework:read
urn:oracle:restframework:update urn:oracle:restframework:delete
urn:oracle:restframework:execute",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d",
    "rel" : "self"
  }, {
    "resourceType" : "urn:oracle:wcps:conductor:scenario",
    "capabilities" : "urn:oracle:restframework:read",
    "href" :
"http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20Worl
d/definition",
    "rel" : "alternate"
  } ]
}
```

68.4.4.5 Deleting a Scenario (DELETE)

XML/JSON

Request:

```
DELETE
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml
```

Response:

```
Status Code:204
Content-Length: 0
X-Powered-By: Servlet/2.5 JSP/2.1
Connection: close
Set-Cookie:
```

```
JSESSIONID=nyKyM2WdGDL6v6Qsgpn335Vgzs5b7vnS417p9dcTgGrlv3736mmm!822326988;
path=/wcps; HttpOnly
Date: Wed, 13 Oct 2010 20:28:45 GMT
```

68.4.4.6 Execute a Scenario (POST)

XML

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/xml
accept: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<parameters>
  <parameter>
    <name>input</name>
    <value>input parameter value</value>
  </parameter>
</parameters>
```

Response:

```
Status Code:200
Content-Length: 348
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=FlFzM2WchrvP59qBJdbSQ1db7TvPQPyRj1HsvJJPCQhKrj7z6zs8!822326988;
path=/wcps; HttpOnly
Content-Type: application/xml
Date: Wed, 13 Oct 2010 20:28:44 GMT

<?xml version="1.0" encoding="UTF-8"?>
<conductorExecutionResults resourceType="urn:oracle:wcps:conductor:genericType">
  <results xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">Hello World with input 'input parameter value'!</results>
</conductorExecutionResults>
```

JSON

Request:

```
POST
http://localhost:7001/wcps/api/conductor/namespaces/Oracle/scenarios/Hello%20World
?projection=details&startIndex=0&itemsPerPage=10
accept-language: en
content-type: application/json
accept: application/json

{
  "parameters" : [ {
    "name" : "input",
    "value" : "input parameter value"
  } ]
}
```

Response:

```
Status Code:200
X-Powered-By: Servlet/2.5 JSP/2.1
Set-Cookie:
JSESSIONID=fhJpM2hf45WSGZl3dwwBnMhCkxPbQwQTF9GLZ15QFc42ZQyVKLcj!822326988;
path=/wcps; HttpOnly
Transfer-Encoding: chunked
Content-Type: application/json
Date: Wed, 13 Oct 2010 20:45:51 GMT

{
  "results" : "Hello World with input 'input parameter value'!",
  "resourceType" : "urn:oracle:wcps:conductor:genericType"
}
```

68.5 Calling Personalization Client Services Using ELs

Using Expression Language (EL) expressions, you can call Personalization services from remote clients such as web pages and Java Server Faces environments. You can, for example, use ELs to apply Personalization scenarios to pages in WebCenter Portal: Spaces. For more information about Personalization ELs, see also Section G.9, "ELs Related to the Personalization Service."

This section contains the following subsections:

- Section 68.5.1, "Calling Personalization Client Services from JSF Pages"
- Section 68.5.2, "Calling Personalization Client Services from JSP Pages"
- Section 68.5.3, "Personalization Context Object Method Reference"
- Section 68.5.4, "Enabling Single Sign-on"

68.5.1 Calling Personalization Client Services from JSF Pages

The Personalization Context is a session-scoped managed bean that is automatically declared and instantiated by the Java Server Faces context, and there for use with the following EL:

```
#{p13nContext}
```

Accessing the Conductor context

Access to remote Conductor services can be accessed in the following way:

```
#{p13nContext.conductor["<UrlOrConnectionName>"]}
```

where ***UrlOrConnectionName*** is a URL to the conductor resource index (for example, `http://localhost:7001/wcps/api/conductor/resourceIndex`), or a connection name specified in `connections.xml` (available only within JDeveloper or WebCenter environments).

If the connection name "`default`" is specified, the first connection name that starts with "`Conductor`" is retrieved. For example:

```
#{p13nContext.conductor['default'].namespaces['myNamespace'].scenario['myScenario'
].results}
```

Accessing the Properties service context

Access to remote Properties Service services can be accessed in the following way:

```
#{p13nContext.properties["<UrlOrConnectionName>"]}
```

where ***UrlOrConnectionName*** is a URL to the Properties Service resource index (for example `http://localhost:7001/wcps/api/property/resourceIndex`), or a connection name specified in `connections.xml` (available only within JDeveloper or WebCenter environments).

If the connection name "`default`" is specified, the first connection name that starts with "`Properties`" is retrieved. For example:

```
#{p13nContext.properties['default'].namespaces['myNamespace'].setDefinitions['MyPr
opertySetDefinition'].sets['MyPropertySet'].results['MyPropertyName']}
```

Configuring the client services for request scope

You may sometimes need to have the Personalization services client use request scope rather than session scope. To do this, configure a custom managed bean in your application's `faces-config` as shown in the example below:

```
<managed-bean>
    <managed-bean-name>myP13nContext</managed-bean-name>
    <managed-bean-class>oracle.wcps.client.PersonalizationContext
 </managed-bean-class>
    <managed-bean-scope>request</managed-bean-scope>
</managed-bean>
```

68.5.2 Calling Personalization Client Services from JSP Pages

The Personalization client can be used with

JSP pages similarly to the way it is used in Java Server Faces pages (see Section 68.5.1, "Calling Personalization Client Services from JSF Pages"). You must, however, register the bean using `<jsp:useBean>` as shown in the following example:

```
<jsp:useBean id="p13nContext" class="oracle.wcps.client.PersonalizationContext"
scope="session"/>
<c:out
value="${p13nContext.conductor['http://weblogic:webl0gic@localhost:7001/wcps/api/c
onductor'].namespaces['default'].scenario['my_scenario_
name'].withInput['question=Does JSP work?'].results}"/>
```

If you want to clear the bean's state, you can invoke it using scriptlet code by placing it after the `<jsp:useBean>` statement:

```
<jsp:scriptlet>((oracle.wcps.client.PersonalizationContext)session.getAttribute("p
13nContext")).reset();</jsp:scriptlet>
```

68.5.3 Personalization Context Object Method Reference

Table 68–8 shows the Personalization context object methods.

Table 68–8 Personalization Context Object Method Reference

Method Name	Description	Example
reset	Clears the state of the context object in which it is invoked.	`#{p13nContext.reset}` `#{p13nContext.conductor['ConductorConnection'].reset}` `#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].reset}` `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].reset}`
update	Updates the property set with new values if bound to a form. This method only applies to the PropertySetContext. **Note:** This method can only called as part of a form-based action.	`#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].update}`
results	Retrieves the results of the context object. For PropertySetContext, it retrieves the property set by name and property set definitions, if they exist. For ScenarioExecutionContext and ParameterizedScenarioExecutionContext, the scenario results are retrieved.	`#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].results}` `#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].results}` `#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].withInput['input1=val1;input2=value2;input3=value3'].results}`
withInput	Executes a scenario by name with input parameters. Input parameters must be in the format: `<paramOne>=<valueOne>;<paramTwo>=<valueTwo>;<paramThree>=<valueThree>` where each parameter name and value is separated by semicolon. This method only applies to ScenarioExecutionContext.	`#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].withInput['input1=val1;input2=value2;input3=value3'].results}`
isError	Determines if an error has occurred in the current context object. Only applies to PropertySetContext and ~ScenarioExecutionContext.	`#{p13nContext.conductor['ConductorConnection'].namespaces[myManagedBean.namespace].scenario[myManagedBean.scenario].isError}` `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].isError}`
errorMessage	Returns the error message if an error has occurred in the current context object. Only applies to PropertySetContext and ~ScenarioExecutionContext.	`#{p13nContext.conductor['ConductorConnection'].namespaces[myManagedBean.namespace].scenario[myManagedBean.scenario].errorMessage}` `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].errorMessage}`

68.5.4 Enabling Single Sign-on

To enable single sign-on, access to the `HttpServletRequest` must be available for both JSF and JSP environments. An implementation of `javax.servlet.Filter` (`PersonalizationFilter`) puts the `HttpServletRequest` on `ThreadLocal` for access within the client as shown in the example below:

```
<filter>
     <description>Personalization Filter for staging client-side
objects.</description>
     <filter-name>PersonalizationFilter</filter-name>
     <filter-class>oracle.wcps.client.PersonalizationFilter</filter-class>
</filter>

<filter-mapping>
     <filter-name>PersonalizationFilter</filter-name>
     <url-pattern>/*</url-pattern>
     <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

This filter is automatically configured in the `wcps-services-client-web-lib`.

Part XII

Completing Your WebCenter Portal Application

Part XII contains the following chapters:

- Chapter 69, "Securing Your WebCenter Portal: Framework Application"
- Chapter 70, "Deploying and Testing Your WebCenter Portal: Framework Application"
- Chapter 71, "Building Multilanguage Portals"

69

Securing Your WebCenter Portal: Framework Application

This chapter describes the security mechanisms and features provided in a WebCenter Portal: Framework application, and how you can use Oracle ADF Security to handle authentication and authorization.

This chapter includes the following sections:

- Section 69.1, "Introduction to WebCenter Portal: Framework Application Security"
- Section 69.2, "Creating an Application Role"
- Section 69.3, "Configuring ADF Security"
- Section 69.4, "Using the Role Manager Task Flow"
- Section 69.5, "Using the Enterprise Role Member Task Flows"
- Section 69.6, "Using the Page Hierarchy Security Editor"
- Section 69.7, "Creating Login Pages and a Login Component"
- Section 69.8, "Creating a Login Portlet"
- Section 69.9, "Adding Portlets to a Login Page"
- Section 69.10, "Creating a Self-Registration Page"
- Section 69.11, "Creating a Reset Password Page"
- Section 69.12, "Configuring Basic Authentication for Testing Portlet Personalization"
- Section 69.13, "Working with External Applications"
- Section 69.14, "Registering Custom Certificates with the Keystore"
- Section 69.15, "Overriding Inherited Security on Portlets and Customizable Components"
- Section 69.16, "Identity Propagation Mechanisms"
- Section 69.17, "Securing Identity Propagation Through WSRP Producers with WS-Security"
- Section 69.18, "Implementing PDK-Java Portlet Security"
- Section 69.19, "Troubleshooting Security Issues"

69.1 Introduction to WebCenter Portal: Framework Application Security

WebCenter Portal: Framework applications are dynamic and often involve input from users in the form of customizations and preferences, and consequently require a flexible security model. The WebCenter security model is based on the ADF security model rather than the more traditional J2EE security model. For more information on ADF security, the section on "Enabling ADF Security in a Fusion Web Application" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

By default, a Framework application is configured with ADF security. A default username and password (weblogic/weblogic1) are created automatically for you, and you can use this username/password combination for testing purposes immediately. For more information about configuring ADF security, see Section 69.3, "Configuring ADF Security."

Default login and logout pages are also provided with the Framework application template.

Framework applications can also use two out-of-the-box security tools: the *Role Manager* task flow, and *Page Hierarchy* security editor. The Role Manager task flow provides pre-defined runtime security administration functionality to define roles and permissions for users. Using the Page Hierarchy security editor provides a quick way to apply inherited and/or delegated security to application pages. These two security tools are described in Section 69.4, "Using the Role Manager Task Flow," and Section 69.6, "Using the Page Hierarchy Security Editor."

69.2 Creating an Application Role

There are three roles available out-of-the box:

- **Administrator** - the default member for the application Administrator role is the enterprise Administrator role, whose default member is `weblogic`
- **AppConnectionManager** - the default member for the AppConnectionManager role is the application Administrator role
- **AppConnectionViewer** - the default member for the AppConnectionViewer role is the authenticated-role

The Administrator role is provided so that you can set up navigation and security for your application, and delegate permissions to other users.

The AppConnectionManager and AppConnectionViewer are roles that are defined for managing and viewing application connections respectively. Typically, application connections are configured and managed using Fusion Middleware Control or WLST commands. However, connections for portlet producers and external applications can be configured through the application's runtime Administration page (or the Role Manager task flow if implemented separately). To manage or view these connections, you must be part of one of these roles. For more information about the default roles, see "Default Application Roles" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

You can add members at runtime for these roles using the Administrator page (or the Role Manager task flow if implemented separately), and in JDeveloper using the JAZN Editor. Out-of-the-box, an administrator will be able to configure external application and portlet producer connections, and any other authenticated user will only be able to view these connections. Similarly, you can add users to the application by selecting the Users tab in the ADF Security Policy editor, clicking the Add (+) icon, and specifying a username and password.

To create an application role:

1. Access the ADF Security Policy editor using any of the following methods:
   - Navigate to the **jazn-data.xml** file in the Application Resources Panel, under the META-INF folder. Right-click **jazn-data.xml** and select **Open** from the context menu. Open the Application Roles tab.
   - Right-click the application name, select **Secure** and then **Application Roles**.
   - From the Application menu, select **Secure** and then **Application Roles**.
2. In the Application Roles editor, click the Add (+) icon and select **New Role**.
3. Enter a name, a display name, and description for the new role.
4. Save the file.

Similarly, you can add users to the application by opening the Users tab in the ADF Security Policy editor, clicking the Add (+) icon, and specifying a username and password.

69.3 Configuring ADF Security

ADF security is configured by default if you created your application using the Framework application template. This section describes the Configure ADF security wizard, which you can use to override the default settings for a Framework application, or if your application does not use the Framework application template (i.e., you did not select the "Configure the application with standard portal features" checkbox when you created your application). This section also describes the grants that are generated when you create a Framework application, and when services are consumed.

This section contains the following subsections:

- Section 69.3.1, "Configuring ADF Security Settings"
- Section 69.3.2, "Automated Security Grants for WebCenter Portal Services"

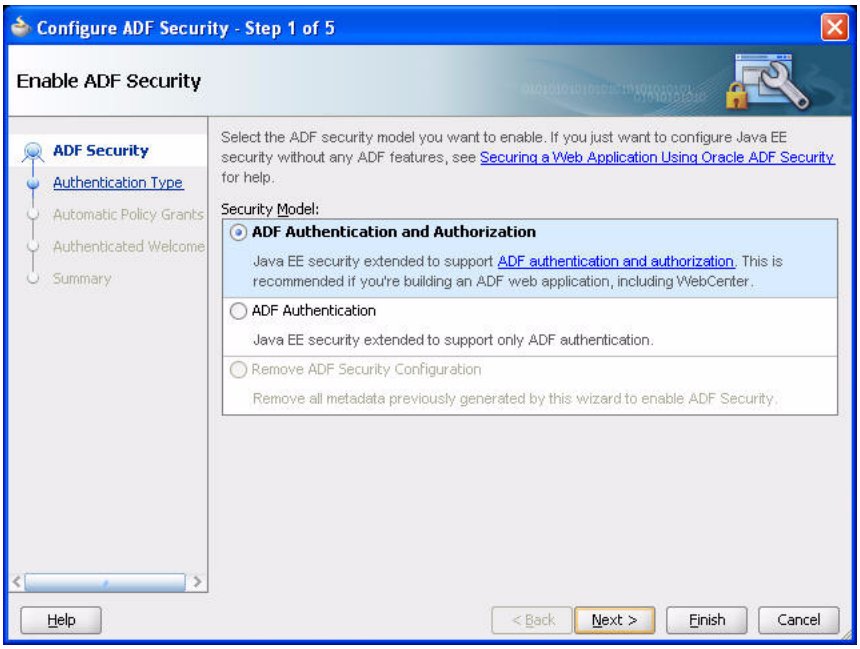
69.3.1 Configuring ADF Security Settings

This section describes the Configure ADF security wizard, which you can use to override the default settings for a Framework application, or if your application does not use the Framework application template (i.e., you did not select the "Configure the application with standard portal features" checkbox when you created your application).

To configure the ADF security settings:

1. Open your portal application.
2. Access the Configure ADF Security Wizard using one of the following methods:
   - From the **Application** menu, select **Secure** and then **Configure ADF Security**.
   - Right-click the application name, select **Secure** and then **Configure ADF Security**.
3. On the Enable ADF Security page, select **ADF Authentication and Authorization**, which is the typical choice for Framework applications.

Figure 69–1 ADF Security Page of the ADF Security Wizard



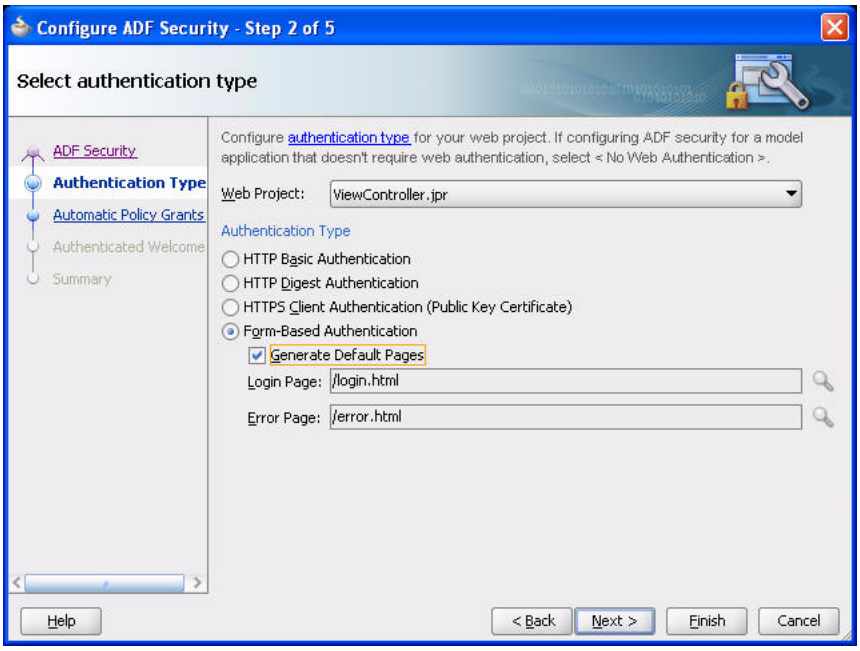
4. Click **Next**.
5. Notice that form-based authentication is selected by default.

   To make the process of securing your application easy, select **Form-Based Authentication** and then the **Generate Default Pages** option, as shown in Figure 69–2.

   Note: Framework applications provide default login and error pages. You should only select the **Generate Default Pages** option for applications not built using the Framework application template.

   Alternatively, if you do not select Generate Default Pages, you can create custom login and error pages. For information about creating custom login pages, see Section 69.7, "Creating Login Pages and a Login Component."

Figure 69–2 Authentication Page of ADF Security Wizard

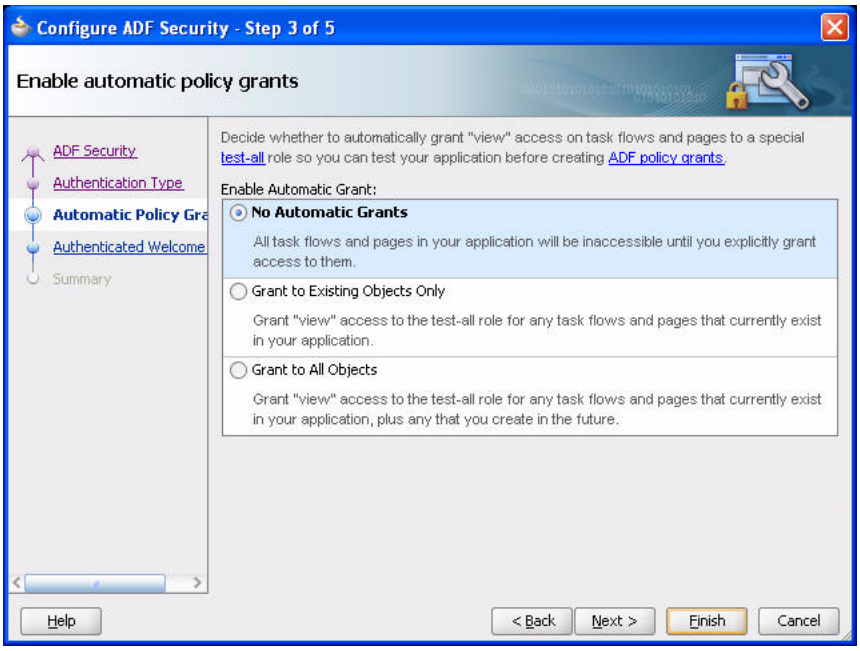


6. Click **Next**.

7. On the Enable automatic policy grants page, shown in Figure 69–3, select whether to make ADF resources public without requiring application developers to first define ADF policy grants. For a standard security implementation, you can select the **No Automatic Grant** option.

   However, in this case, you must individually grant permissions on each page in the application.

Figure 69–3 Automatic Policy Grants Page



When you enable automatic policy grants, a *View* grant for all ADF resources is made to the `test-all` application role, whose default member is the `public` role. This option provides a convenient way to run and test application resources without the restricted access that ADF authorization enforces. You can also disable automatic policy grants to secure all ADF resources by default.

Select **No Automatic Grant** when you want to explicitly configure a policy store to grant access to ADF resources. No automatic grants to the `test-all` application role are granted. However when WebCenter Portal services task flows are consumed in an application, the WebCenter extension automatically adds appropriate grants for these task flows, giving View access to the role `authenticated-role`. For more information about the specific grants provided through task flows, see Section 69.3.2, "Automated Security Grants for WebCenter Portal Services."

Note: By default, `anonymous-user` is granted permission to access services task flows. Anonymous access is also provided to the `resourceViewer` task flow that allows resources to be visible when looking at data through Activity Stream or Search. If you have no public or anonymous access to your functionality, all the grants that are currently given to `anonymous-role`, should be moved to `authenticated-role`. Also, grants made to `authenticated-role` are visible in the source view of the `jazn-data.xml` file.

If the default grants to `anonymous-role` or `authenticated-role` are not appropriate to the application, you can change them by opening the source view of the application's the `jazn-data.xml` file and cutting and pasting the permissions to the appropriate role.

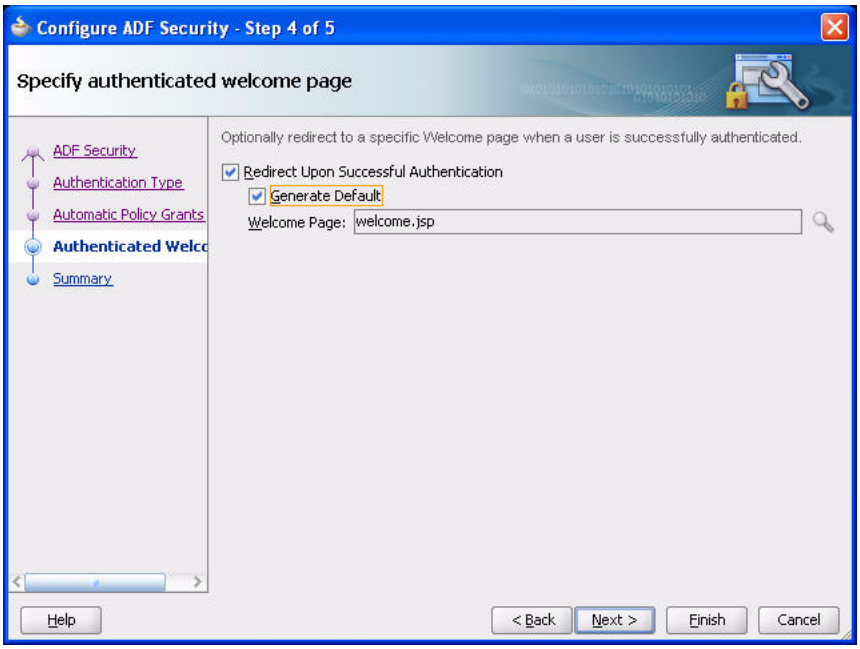
Select **Grant to Existing Objects Only** when you want JDeveloper to grant View privileges to the `test-all` (public) application role and you want this policy to apply to all your application's existing ADF task flows and web pages in the user interface project.

Select **Grant to All Objects** when you want JDeveloper to grant View privileges to the `test-all` application role and you want this policy to apply to all ADF task flows and web pages that developers create in the user interface project. Note that the **Grant to All Objects** option appears when you run the wizard for the first time. Subsequently, the **Grant to New Objects** option appears each time you run the wizard.

8. Click **Next**.
9. Optionally, if you want to display a specific welcome page upon authentication, then select the **Redirect Upon Successful Authentication** option as shown in Figure 69–4.

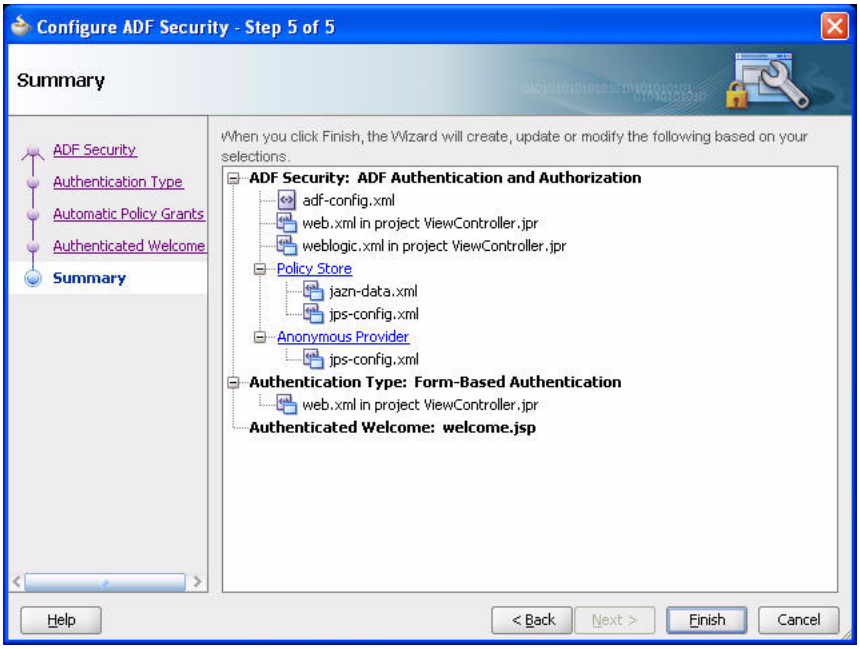
   You can either create a custom welcome page or let the wizard create a default one. To keep the process simple, select **Generate Default**.

Figure 69–4 Specify an Authenticated Welcome Page



10. The Summary page opens, as shown in Figure 69–5.

Figure 69–5 Summary Page



11. Click **Finish**.

It may take several moments for the wizard to complete.

12. On the Security Infrastructure Created dialog, click **OK**.

69.3.2 Automated Security Grants for WebCenter Portal Services

Table 69–1 lists the security grants that are given automatically in a Framework application, configured with standard Portal features.

Note: If you have do not have public or anonymous access to your application, grants that are currently given to `anonymous-role` can be removed using the following WLST command:

```
revokePermission(appStripe="papp1#V2.0",principalClass="oracle.secu
rity.jps.internal.core.principals.JpsAnonymousRoleImpl",
principalName="anonymous-role",
permClass="oracle.adf.controller.security.TaskFlowPermission",
permActions="view", permTarget="/oracle/webcenter/.*")
```

Table 69–1 Automated Security Grants for Framework applications

Role	Grant	Actions	Resource Name
anonymous-role	TaskflowPermission	View	/oracle/webcenter/siteresources/scopedMD/.*
			/oracle/webcenter/portalapp/.*
			/oracle/webcenter/navigationtaskflows/view/pagebreadcrumb-definition.xml#pagebreadcrumb-definition
			/oracle/webcenter/navigationtaskflows/view/pagemenu-definition.xml#pagemenu-definition
			/oracle/webcenter/navigationtaskflows/view/pagetree-definition.xml#pagetree-definition
			/oracle/webcenter/navigationtaskflows/view/.*
			/oracle/webcenter/.*
	RegionPermission	View	oracle.webcenter.siteresources.scopedMD.*
			oracle.webcenter.taskflow.view.ContainerPageDef
			oracle.webcenter.taskflow.view.ViewerPageDef
			oracle.webcenter.taskflowstyle.view.PreviewPageDef
			oracle.webcenter.page.pstemplates.*
			oracle.webcenter.portalapp.pagetemplates.*
			oracle.webcenter.portalapp.pages.loginPageDef
			oracle.webcenter.portalapp.pages.errorPageDef
			oracle.webcenter.portalapp.pages.navigation_rendererPageDef
	HierarchicalResourcePermission	View	serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed
authenticated-role	RegionPermission	View	oracle.webcenter.portalwebapp.pages.adminPageDef
			oracle.webcenter.collab.*
			oracle.webcenter.doclib.*
			oracle.webcenter.tagging.*
	RegionPermission	Grant	oracle.webcenter.taskflow.view.ContainerPageDef
	HierarchicalResourcePermission	Personalize, View	serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed

Table 69–1 (Cont.) Automated Security Grants for Framework applications

Role	Grant	Actions	Resource Name
	CatalogPermission	Create, Edit	/oracle/webcenter/quicklinks/scopedMD/.*
Administrator	HierarchicalResource Permission	Create, Delete, Grant, Manage, Personalize, Update, View	serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed
	WebCenterResource Permission	Manag	oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_contentpresenter_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_datacontrol_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_navigation_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_pagestyle_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_catalog_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_pagetemplate_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_skin_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_taskflow_.*
			oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_taskflowstyle_.*

Table 69–2 lists the grants that are given when a service is consumed. A service is consumed whenever a component (from the Component Palette) or a task flow (from the Resource Palette) belonging to that service is added to a page.

Grants are added automatically to the `jazn-data.xml` file. They can be viewed and edited through the ADF Security Policy Editor. For more information, see Section 69.3.1, "Configuring ADF Security Settings."

Note: When a WebCenter Portal: Framework application is created, the following element:

```
<name>resource=MyApplicationName</name>
```

(where *`MyApplicationName`* is the name of the application you created) is added to the `jazn-data.xml` file. If you modify the `jazn-data.xml` file, be sure that the application name matches the application name under the `jazn-data.xml` file's <policy-store> element. If these two application names do not match, any functionality that relies on WS-security identity propagation will not work.

If ADF security has not been configured for the application, then these grants are not added; however, if security is later configured, grants are added for all services that have been consumed in the application. Common grants are granted whenever any Oracle WebCenter Portal service is consumed.

Table 69–2 Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
Common	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/framework/service/controller/taskflows/resourceViewer.xml#resourceViewer
				/oracle/webcenter/security/peoplepicker/jsf/taskflows/peoplepicker.xml#peoplepicker
				/oracle/webcenter/peopleconnections/personalweb/view/jsf/regions/profile-gallery.xml#profile-gallery
				/oracle/webcenter/peopleconnections/profile/view/jsf/regions/extended/extended-profile.xml#extended-profile
				/oracle/webcenter/peopleconnections/view/jsf/regions/profile-snapshot.xml#profile-snapshot
				/oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-main-view-taskflow.xml#connections-main-view-taskflow
				/oracle/webcenter/peopleconnections/connections/controller/taskflows/connections-mini-view-taskflow.xml#connections-mini-view-taskflow
				/oracle/webcenter/peopleconnections/connections/controller/taskflows/table-of-connections-taskflow.xml#table-of-connections-taskflow
				/oracle/webcenter/peopleconnections/wall/controller/taskflows/WallDetailViewer.xml#WallDetailViewer
				/oracle/webcenter/peopleconnections/wall/controller/taskflows/WallViewer.xml#WallViewer
				/oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosDetailViewer.xml#KudosDetailViewer
				/oracle/webcenter/peopleconnections/kudos/controller/taskflows/KudosMiniViewer.xml#KudosMiniViewer
				/oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-mainview.xml#activity-streaming-mainview
				/oracle/webcenter/activitystreaming/controller/taskflows/activity-streaming-miniview.xml#activity-streaming-miniview
				/oracle/webcenter/peopleconnections/.*
				/oracle/webcenter/activitystreaming/.*
	authenticated-role	Region Permission	View	oracle.webcenter.security.peoplepicker.pageDefs.oracle_webcenter_security_peoplepicker_jsf_pages_PeoplePickerPagePageDef
	authenticated-role	CustomPage Permission	View	oracle.webcenter.framework.service.view.pageDefs.resourceExternalPageDef
	authenticated-role	Profile Permission	Edit, Share, View	/oracle/webcenter/peopleconnections/profile/s8bba98ff_4cbb_40b8_beee_296c916a23ed/.*
	Administrator	WCApp Permission	Manage	application
	Administrator	Profile Permission	Manage	/oracle/webcenter/peopleconnections/profile/s8bba98ff_4cbb_40b8_beee_296c916a23ed/.*

Table 69–2 (Cont.) Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
Announcement	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view
				/oracle/webcenter/collab/announcement/view/taskflows/mini-view-definition.xml#announcement-mini-view
				/oracle/webcenter/collab/announcement/view/taskflows/link-existing-view-definition.xml#link-existing-view-definition
				/oracle/webcenter/collab/announcement/view/taskflows/config-view-definition.xml#announcement-config-view
				/oracle/webcenter/collab/announcement/view/taskflows/announcement-view-definition.xml#announcement-resource-view
				/oracle/webcenter/collab/announcement/view/taskflows/.*
	authenticated-role	Region Permission	View	oracle.webcenter.collab.announcement.view.pageDefs.oracle_webcenter_collab_announcement_view_jsf_pages_AnnouncementLinkExistingViewPageDef
				oracle.webcenter.collab.announcement.view.pageDefs.oracle_webcenter_collab_announcement_view_jsf_pages_AnnouncementLinkResourceViewPageDef
Composer	anonymous-role	Taskflow Permission	View	/oracle/adfinternal/pageeditor/.*
Discussions	anonymous-role	Taskflow Permission	View	/oracle/webcenter/collab/forum/view/taskflows/main-task-flow.xml#forum-main
				/oracle/webcenter/collab/forum/view/taskflows/miniview-task-flow.xml#forum-miniview
				/oracle/webcenter/collab/forum/view/taskflows/popularTopic-task-flow.xml#forum-popularTopic
				/oracle/webcenter/collab/forum/view/taskflows/recentTopic-task-flow.xml#forum-recentTopic
				/oracle/webcenter/collab/forum/view/taskflows/watchedTopic-task-flow.xml#forum-watchedTopic
				/oracle/webcenter/collab/forum/view/taskflows/watchedForum-task-flow.xml#forum-watchedForum
				/oracle/webcenter/collab/forum/view/taskflows/config-task-flow.xml#forum-config-view
				/oracle/webcenter/collab/forum/view/taskflows/link-existing-task-flow.xml#forum-link-existing
				/oracle/webcenter/collab/forum/view/taskflows/link-new-task-flow.xml#forum-link-new
				/oracle/webcenter/collab/forum/view/taskflows/message-task-flow.xml#forum-message
				/oracle/webcenter/collab/forum/view/taskflows/resource-view-task-flow.xml#forum-resource-view
				/oracle/webcenter/collab/forum/view/taskflows/scope-config-task-flow.xml#forum-scope-config-view
				/oracle/webcenter/collab/forum/view/taskflows/.*

Table 69–2 (Cont.) Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
	authenticated-role	Region Permission	View	oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ForumLinkExistingViewPageDef
				oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ForumLinkNewViewPageDef
				oracle.webcenter.collab.forum.view.pageDefs.oracle_webcenter_collab_forum_view_jsf_pages_ForumLinkResourceViewPageDef
Documents	anonymous-role	Taskflow Permission	View	/oracle/webcenter/doclib/view/jsf/taskflows/presenter/contentPresenter.xml#doclib-content-presenter
				/oracle/webcenter/doclib/view/jsf/taskflows/docListViewer.xml#doclib-document-list-viewer
				/oracle/webcenter/doclib/view/jsf/taskflows/docViewer/docInfo.xml#doclib-document-information
				/oracle/webcenter/doclib/view/jsf/taskflows/explore/explorer.xml#doclib-explorer
				/oracle/webcenter/doclib/view/jsf/taskflows/treeNav/treeNavigator.xml#doclib-navigator
				/oracle/webcenter/doclib/view/jsf/taskflows/docViewer/documentViewer.xml#doclib-document-viewer
				/oracle/webcenter/doclib/view/jsf/taskflows/folderViewer/folderView.xml#doclib-folder-viewer
				/oracle/webcenter/doclib/view/jsf/taskflows/mainView.xml#doclib-document-library
				/oracle/webcenter/doclib/view/jsf/taskflows/miniProperties/miniProperties.xml#doclib-mini-properties
				/oracle/webcenter/doclib/view/jsf/taskflows/recentDocuments.xml#doclib-recent-documents
				/oracle/webcenter/doclib/view/jsf/taskflows/richTextEditor/editor.xml#doclib-richtext-editor
				/oracle/webcenter/doclib/view/jsf/taskflows/upload/uploader.xml#doclib-upload-document
				/oracle/webcenter/doclib/view/jsf/taskflows/versionHistory/history.xml#doclib-version-history
				/oracle/webcenter/blog/view/jsf/taskflows/blogDigestViewer/blog-main-view.xml#blog-main-view
				/oracle/webcenter/doclib/view/jsf/taskflows/.*
				/oracle/webcenter/blog/view/jsf/taskflows/.*
	authenticated-role	Region Permission	View	oracle.webcenter.doclib.view.jsf.*
				oracle.webcenter.blog.view.jsf.*
External Application	anonymous-role	Taskflow Permission	View	/oracle/adfinternal/extapp/view/fragments/extapp-credential-provisioning-taskflow.xml#extapp-credential-provisioning-taskflow
				/oracle/adfinternal/extapp/view/fragments/extapp-change-password-taskflow.xml#extapp-change-password-taskflow

Table 69–2 (Cont.) Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
Links	anonymous-role	Taskflow Permission	View	/oracle/webcenter/relationship/view/jsf/resources/links-detail.xml#links-detail
				/oracle/webcenter/relationship/view/jsf/resources/links-detail-popup.xml#links-detail-popup
				/oracle/webcenter/relationship/view/jsf/resources/links-to-service.xml#link-to-service
				/oracle/webcenter/relationship/view/jsf/resources/links-picker-popup.xml#links-picker-popup
				/oracle/webcenter/relationship/url/view/jsf/taskflows/linkToUrl.xml#link-to-url
	Administrator	Relationship Permission	Manage	*s8bba98ff_4cbb_40b8_beee_296c916a23ed/.*
Lists	anonymous-role	Taskflow Permission	View	/oracle/webcenter/list/view/jsf/regions/main-view-task-flow.xml#main-view-task-flow
				/oracle/webcenter/list/view/jsf/regions/list-instance-view-task-flow.xml#list-instance-view-task-flow
				/oracle/webcenter/list/view/jsf/regions/.*
	authenticated-role	Region Permission	View	/oracle/webcenter/list/templates/lists/.*
	Administrator	List Permission	View	/oracle/webcenter/list/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/lists/.*
Mail	anonymous-role	Taskflow Permission	View	/oracle/webcenter/collab/mail/view/jsf/regions/compose-task-flow.xml#mail-compose-view
				/oracle/webcenter/collab/mail/view/jsf/regions/content-view-definition.xml#mail-content-view
				/oracle/webcenter/collab/mail/view/jsf/regions/dl-config-definition.xml#dl-config-view
				/oracle/webcenter/collab/mail/view/jsf/regions/mini-view-definition.xml#mail-mini-view
				/oracle/webcenter/collab/mail/view/jsf/regions/.*
	authenticated-role	Region Permission	View	oracle.webcenter.collab.mail.view.pageDefs.oracle_webcenter_collab_mail_view_jsf_pages_ComposeviewPageDef
				oracle.webcenter.collab.mail.view.pageDefs.oracle_webcenter_collab_mail_view_jsf_pages_ContentViewPageDef
Notifications	anonymous-role	Taskflow Permission	View	/oracle/webcenter/notification/view/jsf/regions/SubscriptionPreferences.xml#SubscriptionPreferences
				/oracle/webcenter/notification/view/jsf/regions/SubscriptionsViewer.xml#SubscriptionsViewer

Table 69–2 (Cont.) Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
Page	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/page/view/jsf/fragments/page-create-page.xml#page-create-page
				/oracle/webcenter/page/view/jsf/fragments/page-doc-prop-panel-definition.xml#page-doc-prop-panel-definition
				/oracle/webcenter/page/view/jsf/fragments/page-doc-sec-panel-definition.xml#page-doc-sec-panel-definition
				/oracle/webcenter/page/view/jsf/fragments/golink-prop-panel-definition.xml#golink-prop-panel-definition
	authenticated-role	Region Permission	Create	oracle_webcenter_page_scopedMD_s8bba98ff_4cbb_40b8_beee_296c916a23ed_.*
	Administrator	Region Permission	Customize, Edit, Grant, Personalize, View	oracle_webcenter_page_scopedMD_s8bba98ff_4cbb_40b8_beee_296c916a23ed_.*
Polls	anonymous-role	Taskflow Permission	View	/oracle/webcenter/collab/survey/view/jsf/taskflows/list-surveys-definition.xml#list-surveys
				/oracle/webcenter/collab/survey/view/jsf/taskflows/take-survey-definition.xml#take-survey
				/oracle/webcenter/collab/survey/view/jsf/taskflows/quick-poll-definition.xml#quick-poll
				/oracle/webcenter/collab/survey/view/jsf/taskflows/view-results-definition.xml#view-results
				/oracle/webcenter/collab/survey/view/jsf/taskflows/take-polls-definition.xml#take-polls
				/oracle/webcenter/collab/survey/view/jsf/taskflows/empty-definition.xml#empty-definition
				/oracle/webcenter/collab/survey/view/jsf/taskflows/.*
Recent Activities	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/recentactivity/controller/taskflows/recent-activities.xml#recent-activities
				/oracle/webcenter/recentactivity/controller/taskflows/.*
RSS	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/rssviewer/view/jsf/fragments/RSSViewerTaskFlow.xml#RSSViewerTaskFlow
				/oracle/webcenter/rssviewer/view/jsf/fragments/.*

Table 69–2 (Cont.) Automated Security Grants for WebCenter Portal Services

Service	Role	Grant	Actions	Resource Name
Search	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/search/controller/taskflows/searchResults.xml#search-view
				/oracle/webcenter/search/controller/taskflows/localToolbarSearch.xml#search-toolbar
				/oracle/webcenter/search/controller/taskflows/preferences.xml#search-preferences
				/oracle/webcenter/search/controller/taskflows/allSavedSearches.xml#all-saved-searches
				/oracle/webcenter/search/controller/taskflows/simpleSearchResults.xml#search-simple-view
				/oracle/webcenter/search/controller/taskflows/customize.xml#search-customize
				/oracle/webcenter/search/controller/taskflows/.*
Tags	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/tagging/controller/taskflows/related-links.xml#tagging-related-links
				/oracle/webcenter/tagging/controller/taskflows/launch-dialog.xml#tagging-launch-dialog
				/oracle/webcenter/tagging/controller/taskflows/tagging-personal-view.xml#tagging-personal-view
				/oracle/webcenter/tagging/controller/taskflows/tag-selection.xml#tag-selection
				/oracle/webcenter/tagging/controller/taskflows/related-resources.xml#related-resources
				/oracle/webcenter/tagging/controller/taskflows/tag-center-task-flow.xml#tag-center
				/oracle/webcenter/tagging/controller/taskflows/tag-center-related-tags.xml#tag-center-related-tags
				/oracle/webcenter/tagging/controller/taskflows/tag-center-related-users.xml#tag-center-related-users
	authenticated-role	Region Permission	View	oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_launch_dialogPageDef
				oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_centerPageDef
				oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_resourcesPageDef
				oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_tagsPageDef
				oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_center_related_usersPageDef
				oracle.webcenter.tagging.view.pageDefs.oracle_webcenter_tagging_view_jsf_fragments_tag_center_tag_selectionPageDef
Worklist	anonymous-role	TaskFlow Permission	View	/oracle/webcenter/worklist/view/jsf/taskFlowDefs/worklist.xml#worklist
				/oracle/webcenter/worklist/view/jsf/taskFlowDefs/.*

69.4 Using the Role Manager Task Flow

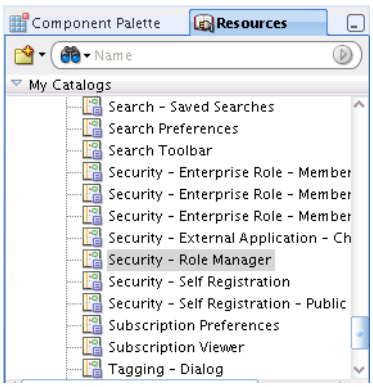
The Role Manager task flow lets administrators manage application roles, its memberships, and associate service permissions for the role. The Role Manager task flow is available out-of-the-box from the runtime Administration page's Security tab for applications built using the Framework application template. For information on how to use the Role Manager runtime to manage and configure users and application roles, see "Managing Users and Application Roles" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To prevent customizations from being overwritten during updates, the source for the Administration page should not be customized. You can, however, choose to write your own administration page if the default administration page is not sufficient, in which case you can use the Role Manager task flow as part of your application's runtime security administration. Follow the steps below to add it a JSF or JSP page and include it in your application.

To add the Role Manager task flow to your Framework application:

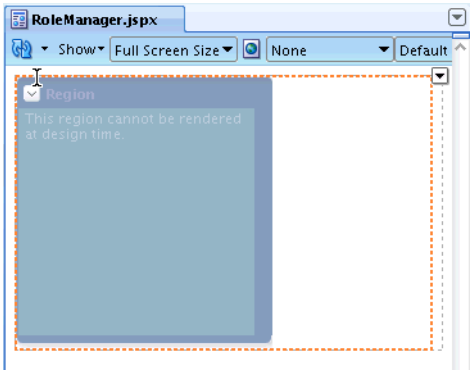
1. Create a new application choosing **Framework application** as the application type.
2. Create the application adding any project technology libraries that the application may require. Do not uncheck the **Configure the application with standard Portal features** check box.
3. Create a new JSF or JSP page.
4. From the Resource Palette, expand WebCenter Portal - Services Catalog, and from the list of Task Flows, select the `Security - Role Manager` task flow.

Figure 69–6 Resource Palette - WebCenter Portal - Services Catalog



5. Drag the Role Manager task flow onto the page you created as a region.

Figure 69–7 Role Manager Task Flow



6. Build the application, log in as an administrator, and navigate to the page containing the role Manager task flow.

69.5 Using the Enterprise Role Member Task Flows

The Enterprise Role Member task flows provide additional task flows to the Role Manager task flow that you can add to a page to monitor enterprise users and roles. The members shown are both roles and users, with the result shown in a tree structure. Roles can be further expanded to see their members.

To add the task flows, follow the same steps as for the Role Manager task flow (see Section 69.4, "Using the Role Manager Task Flow"), substituting one of the task flows described below. Note that all of the Enterprise Role Members task flows take a single mandatory parameter which is the enterprise role name.

- **Security - Enterprise Role Members**

  This task flow shows a list of the members for a specified role.

- **Security - Enterprise Role - Members Search**

  This task flow does a nested search of members using a pattern for the specified role and shows the list of members.

- **Security - Enterprise Role - Members Viewer**

  This task flow shows the Members and Search tabs that are shown for the Enterprise Role Members and Members Search task flows in a single page.

69.6 Using the Page Hierarchy Security Editor

This section describes how to use page hierarchy security to provide inherited and delegated security for WebCenter Portal: Framework application pages. It includes the following sections:

- Section 69.6.1, "Introduction to Page Hierarchy Security"
- Section 69.6.2, "Building a Page Hierarchy"

69.6.1 Introduction to Page Hierarchy Security

This section describes the page hierarchy model and how page hierarchy security works within a WebCenter Portal: Framework application.

This section includes the following subsections:

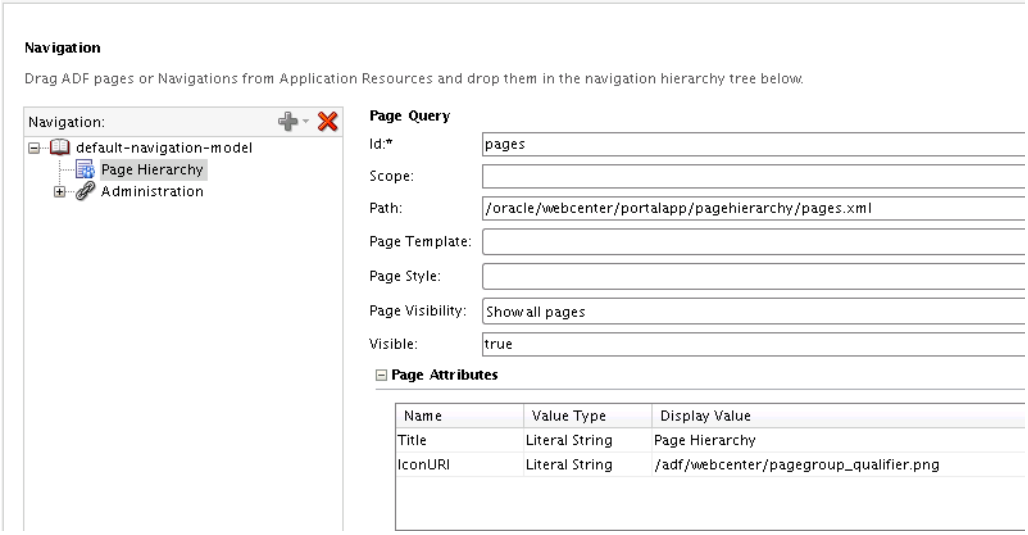
- Section 69.6.1.1, "Page Hierarchy Model"
- Section 69.6.1.2, "Page-level and Page Hierarchy Security"
- Section 69.6.1.3, "Inherited and Delegated Security"

69.6.1.1 Page Hierarchy Model

A page hierarchy has a default security policy that is set at the root of the hierarchy. Unless explicitly overridden, this policy applies to all pages underneath. At any page in the hierarchy, it is possible to override the inherited policy with a new overriding policy, allowing for delegated administration and modifying the policy at the subordinate pages. Once the policy is overridden for a particular page, the children of that page, by default, will inherit the new policy. The policy established at any node is a list of grants indicating what permissions each of the grantees has on that page and its children, if any.

When you create a WebCenter Portal: Framework Application, a default navigation model is created for you that also contains the root node of a page hierarchy. These two structures work independently and collectively to provide the navigation flow and secured access to pages for your application. Figure 69–8 shows the default navigation model (`default-navigation-model.xml`), which includes a node for the default page hierarchy (`pages.xml`).

Figure 69–8 Default Navigation Model



You can add additional navigation models to, for example, provide different navigation views for specific user groups. For each navigation you can also add one or more page hierarchy nodes to provide a more granular navigation and security model. For more information about creating new navigation models, see Section 11.2, "Creating a Navigation Model." For more information about adding page hierarchies to a navigation, see Section 11.3.5, "How to Add a Page Hierarchy to a Navigation Model."

Although there is only a single page hierarchy within your application, any node within that hierarchy can be used as the root node in a navigation model. For example, you might have a page hierarchy that contains top level nodes for Human Resources, Support, Sales, Administration, and so forth. Each of these nodes can be added to the navigation model separately to create navigation only to that section of the page hierarchy.

Figure 69–9 Page Hierarchy Root Node Permissions

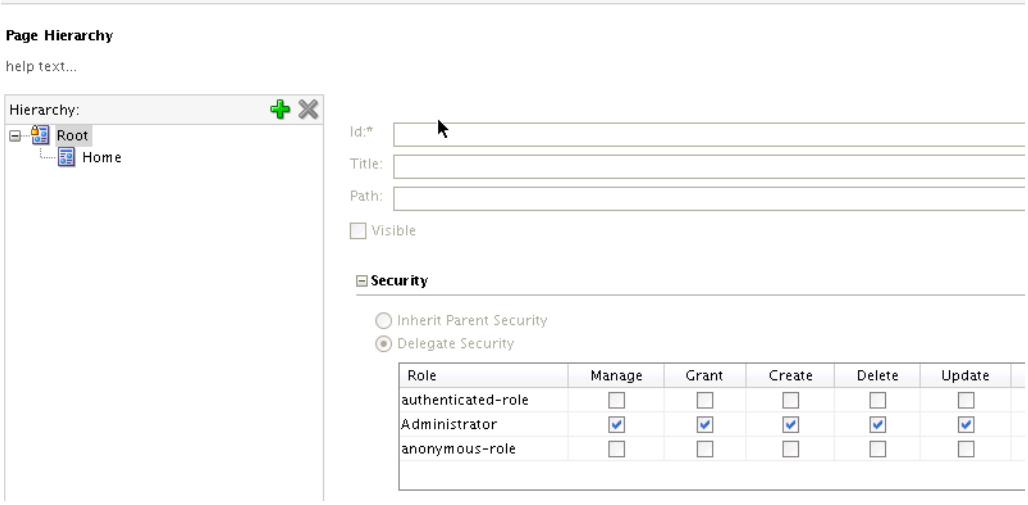


Figure 69–9 shows the page hierarchy root node and the associated permissions. From the root node you can start building your page hierarchy by adding or dragging pages into it. Note that the "Manage" permission appears only on the root node, and only for the administrator. This allows the administrator to delegate security and have full access to all pages within the hierarchy regardless of any security overrides downstream. In short, a user with "manage" permission on the root node is a "super admin" and has all access to all pages in the entire hierarchy.

69.6.1.2 Page-level and Page Hierarchy Security

A page hierarchy provides an alternative security mechanism to page-level security for your Framework application. Although you can add pages to your application and provide security for them without making them a part of a page hierarchy, a page hierarchy provides a convenient and more manageable way to secure the content for large sites. By simply adding a hierarchy node to the hierarchy root or home node and dragging and dropping pages into it, you can very quickly provide basic security for them.

> **Note:** Although page-level security can co-exist with page hierarchy security, you cannot apply both page-level and page hierarchy security to the same page. Note also that if you add a page to a hierarchy node, all previous page-level permissions are purged.

69.6.1.3 Inherited and Delegated Security

The page hierarchy model uses a node structure that provides cascaded, role-based security where each child node inherits its parent's security permissions. By adding pages to a node you can quickly configure security for your application pages.

To be able to perform any action at a particular node, you need to have the required permission at the node or its immediate parent that delegates (overrides) security. However in order to view a page, you need to have view access for all pages up the hierarchy right till the root page.

As well as providing basic security inherited from a node's parent, you can also apply delegated security to child nodes to limit or override the security that would otherwise

have been inherited. These aspects of page hierarchy security are described in Section 69.6.2, "Building a Page Hierarchy."

> **Note:** Inherited security is cascaded through a root node that can initially only be populated by an administrator. For other users to be able to create and delegate permissions the administrator must first delegate those permissions. In all cases, permissions are only cascaded down in the tree (that is, a user cannot change settings for a parent node). A user can also only delegate the rights they have themselves or a subset of those rights to another user.

69.6.2 Building a Page Hierarchy

This section explains how to use page hierarchy security to automatically cascade a parent page's role-based security settings to its child pages, and how to "delegate" security for individual page nodes.

This section contains the following subsections:

- Section 69.6.2.1, "Adding a Page Hierarchy to a Navigation"
- Section 69.6.2.2, "Adding Pages to a Page Hierarchy"
- Section 69.6.2.3, "Delegating Security"

69.6.2.1 Adding a Page Hierarchy to a Navigation

In addition to the default page hierarchy in the default navigation model, you can add page hierarchy nodes from the default page hierarchy to the navigations within your application as root nodes. For more information about adding a page hierarchy to a navigation, see Section 11.3.5, "How to Add a Page Hierarchy to a Navigation Model."

69.6.2.2 Adding Pages to a Page Hierarchy

This section describes how to add JSF or JSP pages to a page hierarchy.

> **Note:** To avoid issues with conflicting inherited security models, a page can only appear once in the page hierarchy.

To add a JSF or JSP page to a page hierarchy:

1. Open the page hierarchy editor in JDeveloper:
   - Right-click the `/oracle/webcenter/portalapp/pagehierarchy` folder

     or
   - Right-click any `.jspx` page under `/oracle/webcenter/portalapp` and select **Edit Page Hierarchy** from the context menu
2. Select the node to which you want to add pages, and either:
   - Click the **Add page** icon and use the browser to locate and add the page

     or
   - Drag and drop the page from the Navigator under the appropriate node
3. Continue by adding additional pages as required, or use the **Remove Page** icon to delete any unwanted pages.

69.6.2.3 Delegating Security

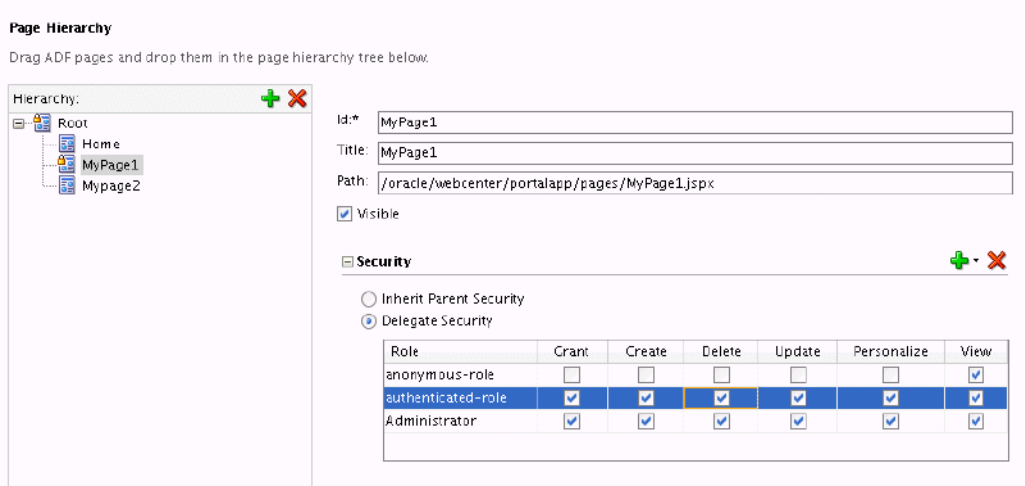
This section describes how to override inherited security by delegating permissions for a node and its child pages. You can also delegate security for pages using the runtime Administration page. For more information about delegating security using the runtime Administration page, see "Managing Application Roles and Permissions" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To override inherited security with delegated security:

1. Open the page hierarchy (`pages.xml`) in JDeveloper.
2. Select the node at which to begin overriding inherited security.
3. Check or uncheck the **Visible** checkbox to specify whether the page or resource is displayed. Check the checkbox if you want the resource to be displayed to all users at all times. Uncheck the checkbox to hide the resource from all users at all times.
4. Select **Delegate Security**.

   The Security options for delegating security displays (see Figure 69–10).

Figure 69–10 Page Hierarchy - Delegate Security



5. Check or uncheck permissions for the roles for this page and its child pages.
6. To specify permissions for another available user role, click the **Add** icon (**+**) and select the desired role. A new row will be created in the table for this role. To remove the permissions for an existing role, select the corresponding row in the table and click the **Delete** (**X**) icon.

> **Note:** The role names in the Add list correspond to the application roles that have been defined in the policy store for the application. Delegating security is limited to only application roles at design time. You can, however, add user and enterprise roles at runtime using the runtime Administration page, as described in "Managing Application Roles and Permissions" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

69.7 Creating Login Pages and a Login Component

You can add login and logout links to your public welcome page or other page using a login backing bean (`LoginBackingBean`) so that users can explicitly log in and out while they are in the application. The bean provides methods that can be used by the user to log in and log out.

To use the bean, create a simple page and bind it to the bean to do the login and logout as described in the following steps:

1. Create a JSF page.
2. Add the `LoginBackingBean` to the `faces-config.xml` or `taskflow.xml` file.
3. Create a page with user name, password, login and logout links.
4. Bind the user name and password to the bean's user name and password.
5. Map the login and logout links to the `doLogin` and `doLogout` method respectively as shown in the example below:

```
          <af:subform id="pt_sf1" defaultCommand="pt_logincb"
                      rendered="#{attrs.showLogin and
!securityContext.authenticated}">
               <af:panelFormLayout id="pt_pfl1">
                 <af:panelLabelAndMessage id="pt_plam1" label="User Name"
                                          styleClass="NoLabelWrap"

labelStyle="font-size:small;color:white;">
                   <af:inputText id="pt_it1" simple="true"
                                 value="#{o_w_s_l_LoginBackingBean.userName}"
                                 columns="15"/>
                 </af:panelLabelAndMessage>
                 <af:panelLabelAndMessage id="pt_plam2" label="Password"
                                          styleClass="NoLabelWrap"

labelStyle="font-size:small;color:white;">
                   <af:inputText id="pt_it2" simple="true"
                                 value="#{o_w_s_l_LoginBackingBean.password}"
                                 columns="15" secret="true"/>
                 </af:panelLabelAndMessage>
               </af:panelFormLayout>
               <af:spacer width="3" height="3" id="pt_s2"/>
               <af:panelGroupLayout id="pt_pgl14" layout="horizontal"
                                    halign="end">
                 <af:commandLink id="pt_logincb" text="Login"
                                 action="#{o_w_s_l_LoginBackingBean.doLogin}"
                                 inlineStyle="font-size:small;color:white;"/>
                 <af:spacer id="pt_s3" width="5px"/>
               </af:panelGroupLayout>
             </af:subform>
```

The login and logout methods return values that can be mapped to actions and used in the navigation rules of `faces-config.xml` after a successful login and logout. The following entry in `faces-config.xml` is seeded by default in a Framework application:

```
<navigation-rule>
  <from-view-id>*</from-view-id>
  <navigation-case>
    <from-outcome>login_success</from-outcome>
```

```
    <to-view-id>/pages_home</to-view-id>
    <redirect/>
  </navigation-case>
  <navigation-case>
    <from-outcome>logout_success</from-outcome>
    <to-view-id>/pages_home</to-view-id>
    <redirect/>
  </navigation-case>
</navigation-rule>
```

This defines where to go to after a successful login and logout (in this case, for both login and logout the user is returned to the home page). If you want to alter that behavior, you can edit these rules in `faces-config.xml` to, for example, redirect the user to a different page.

For reference information about other configuration elements you can use in the `faces-config.xml` file, see "ADF Faces Configuration" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework.*

69.8 Creating a Login Portlet

Refer to the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for information about creating a login portlet for your application.

69.9 Adding Portlets to a Login Page

The *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* describes how you can create an ADF Faces-based login page for your application. In this section, you will add some portlets to such a login page so that the login page becomes indistinguishable from the other pages in your WebCenter Portal: Framework application.

Make sure your portlet producers have been registered before proceeding. See Section 64.2, "Registering Portlet Producers with a WebCenter Portal: Framework Application" for details.

To add portlets to the login page, perform the following steps:

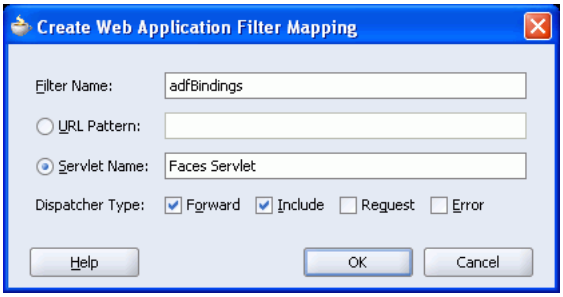
1. Drag a **PanelCustomizable** onto the `h:form` tag that is inside the first `cust:panelCustomizable` tag you added to the login page.
2. From the Component Palette, select **RichTextPortlet Producer**, then select the Rich Text portlet from the list and drag it onto the `PanelCustomizable` component.
3. From the Component Palette, select **ADF Faces Core** and drag an **ObjectSeparator** below the Rich Text portlet on the `PanelCustomizable` component.
4. From the Component Palette, select **OmniPortlet Producer**, then select the OmniPortlet from the list and drag it onto the `PanelCustomizable` component.
5. Save the page.

Because the login page is called from the container as part of the login process, the request must be forwarded to the ADF binding filter to establish the appropriate portlet and security context. To do this, you must configure a mapping for the ADF Binding filter in the `web.xml` file. To do this, perform the following steps:

1. In the Applications Navigator, expand the WEB-INF node, right-click **web.xml** and select **properties** to open the property palette.

2. Select **Filter Mappings** in the left panel and click **add** to define a new mapping for the `adfBindings` Filter. This displays the Create Web Application Filter Mapping dialog box.
3. Specify **adfBindings** for the filter name and click the Servlet Name option and specify **Faces Servlet** as the servlet name. Ensure that the Forward and Include dispatcher types are selected as shown in Figure 69–11.

Figure 69–11 Create Web Application Filter Mapping Dialog Box



4. Click **OK**.

Follow the steps in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* to complete the creation of the login page.

69.10 Creating a Self-Registration Page

You can create a self-registration page using the WebCenter Portal: Spaces self-registration task flow, or build a custom self-registration page based on a provided sample that shows how to use the user and role APIs provided in `CreateUserHelper`. This type of self-registration provides a way for public users to create their own login and password for your application.

You can also invite users who may, for example, be outside your organization to self-register using the Spaces public invitation task flow. For more information about these two types of self-registration pages and their runtime behavior, see the chapter on "Enabling Self-Registration" in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

This section contains the following subsections:

- Section 69.10.1, "Integrating the Spaces Self-Registration Task Flow"
- Section 69.10.2, "Building a Custom Self-Registration Page"

69.10.1 Integrating the Spaces Self-Registration Task Flow

You can integrate the self-registration page provided in Spaces in a Framework application as described below. For information about the runtime behavior of the task flow, see "Enabling Anyone to Self-Register" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

To consume the Spaces self-registration task flow:

1. Open or create a Framework application in JDeveloper.
2. Create a JSPX page.
3. From the Resource Palette, expand the task flows and drag and drop the self-registration task flow onto the page.

4. Create a mail connection and an external application with public credentials and connect the external application to the mail connection. Make this the mail default connection.
5. Run the page with the self-registration task flow to see the runtime self-registration page.

69.10.2 Building a Custom Self-Registration Page

The example self-registration page described below provides a simple page that accepts the user's `firstname`, `lastname`, `mailid`, `login name` and `password`. Use the sample files as a starting point for use within your local environment, or build your own self-registration page and logic using the user and role APIs provided in `CreateUserHelper`. You can use the property set of the user profile attributes provided in the sample and add more attributes to the user as required.

To create the sample self-registration page:

1. Unzip the security samples files in `security-samples.zip` found in the `webcenter/customportal` directory in the WebCenter extension bundle.
2. Copy the `CreateUserBean.java` and `CreateUserHelper.java` files into the application sources folder for your Framework application.
3. Copy the `TestCreateUserPage.jspx` file into the public-html folder.
4. Edit the `TestCreateUserPage.jspx` file to suit the local environment and local requirements.
5. Run the `TestCreateUserPage.jspx` page to view the self-registration page.

69.10.3 Creating a Self-Registration Invitation Page

You can create a public invitation to self-register page in your Framework application using the Spaces public invitation task flow. For information about the runtime behavior of the task flow, see "Enabling Self-Registration by Invitation Only" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

To consume the self-registration invitation task flow:

1. Open or create a Framework application in JDeveloper.
2. Create a JSPX page.
3. From the Resource Palette, expand the task flows and drag and drop the self-registration invitation task flow onto the page.
4. Grant the following `CredentialAccessPermission` in `jazn-data.xml` to `oracle.webcenter.framework.view/-`

   The `jazn-data.xml` file should like like the following:

   ```
   <permission>
   <class>oracle.security.jps.service.credstore.CredentialAccessPermission</class>
   <name>context=SYSTEM,mapName=o.webcenter.security.selfreg,keyName=o.webcenter.s
   ecurity.selfreg.hmackey</name>
   <actions>read,write</actions>
   </permission>
   ```

5. Run the page with the self-registration task flow to see the runtime self-registration invitation page.

69.11 Creating a Reset Password Page

The following example provides a simple reset password page that accepts a username and new password. Use the sample files as a starting point for use within your local environment, or build your own reset password page and logic using the user and role APIs provided in `ResetPasswordHelper` to change the password.

To create the sample reset password page:

1. Unzip the security samples files in `security-samples.zip` found in the `webcenter/customportal` directory in the WebCenter extension bundle.
2. Copy the `ResetPasswordBean.java` and `ResetPasswordHelper.java` files into the application sources folder for your Framework application.
3. Copy the `TestResetPasswordPage.jspx` file into the public-html folder.
4. Edit the `TestResetPasswordPage.jspx` file to suit the local environment and local requirements.
5. Run the `TestResetPasswordPage.jspx` page to view the reset password page.

69.12 Configuring Basic Authentication for Testing Portlet Personalization

Portlet personalizations are tied to particular, authenticated users. Hence, when running a portlet that has an Edit mode, the Personalize option in the portlet's dropdown menu only appears to authenticated users of the application. Anonymous or public users will not have the option to personalize the portlet. If you are a developer creating portlets and pages, then you may want to quickly test the Edit mode of your portlet without creating a complete security model for your application. To perform this sort of testing, you can easily configure some very basic authentication for your application and then remove it when you have finished testing:

Note: This procedure is useful for any portlet that has an Edit mode (Omniportlet, Web Clipping, JPS, and PDK-Java).

---

1. Create a user `sking` and the role `manager`. See the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for information about creating users and roles.
2. Secure your application using the ADF Security Wizard. See the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* for the steps to be performed. On the Login page of the wizard, select **HTTP Basic Authentication (RFC 2617)**. This specifies that the application will use basic authentication.
3. Run the page in the integrated WLS and log in as a valid user and test your portlet's edit mode.

When you are done testing your portlet's Edit mode, you can quickly remove this test security by do the following:

1. In the Applications Navigator, click the project that contains a page with the portlet you want to test.
2. From the **Tools** menu, choose **ADF Security Wizard**.
3. If the Welcome page appears, then click **Next**.

4. Choose **Remove All ADF Security Settings**.
5. Click **Next** until you come to the Finish page of the wizard. Click **Finish**. The security is removed. If you want to ensure that the security has been removed, then exit your browser and rerun the application. When you access the page, you are not prompted to login and the personalize option no longer available from the portlet's dropdown menu.

69.13 Working with External Applications

Oracle WebCenter Framework defines an external application as any application that implements its own authentication process. That is, an application that does not take part in the WebCenter Framework application's single sign-on process. In some cases, the identity management solution may be the same, but the authentication process can be different.

When WebCenter Framework Service interacts with an application that handles its own authentication, you can associate that service with an external application to allow for credential provisioning. Therefore, the use of an external application definition provides a means of accessing content from these independently authenticated applications.

To replicate a single sign-on experience from the end user's perspective, the external application service captures the username and password, and any other credentials for the external application, and supplies it to the WebCenter service requiring it. The WebCenter service then uses this and logs in on behalf of the end user. This username and password combination is securely stored in a credential store configured for the WebLogic domain where the application is deployed.

The user provides login credentials when prompted, and these credentials are mapped to the Framework application user and stored in the credential store configured for the domain. The credential store subsequently supplies that information during authentication to the external application. Unless the external application's credentials change, the user supplies the credentials only once as the mapped information is read from the credential store for future requests.

> **Note:** When logging in to an external application, if you clear the **Remember My Login Information** checkbox, then the credentials provisioned for that user session are lost if there is a failover in a high availability (HA) environment. You are prompted to specify the credentials again if you try to access the external application content in the same user session.

The external applications that are to be used by the WebCenter Portal: Framework application can be specified before deployment through a wizard in Oracle JDeveloper, or post-deployment through the application server management interfaces (WebLogic Scripting Tool and Oracle Enterprise Manager). See *Oracle Application Server Administrator's Guide* for more information.

This section contains the following:

- Section 69.13.1, "Using External Applications"
- Section 69.13.2, "Supplying User Credentials"
- Section 69.13.3, "Managing External Applications"

69.13.1 Using External Applications

The WebCenter External Application Service provides a way for mapped user identities to be passed to a web application that requires its own authentication. The support for external applications and credential mapping provided by the WebCenter External Application Service can be used to set up a secured service connection and to provide a seamless automated single sign-on experience for the user.

This is described in the following sections:

- Section 69.13.1.1, "Secured Service Connections"
- Section 69.13.1.2, "Automated Single Sign-On"

69.13.1.1 Secured Service Connections

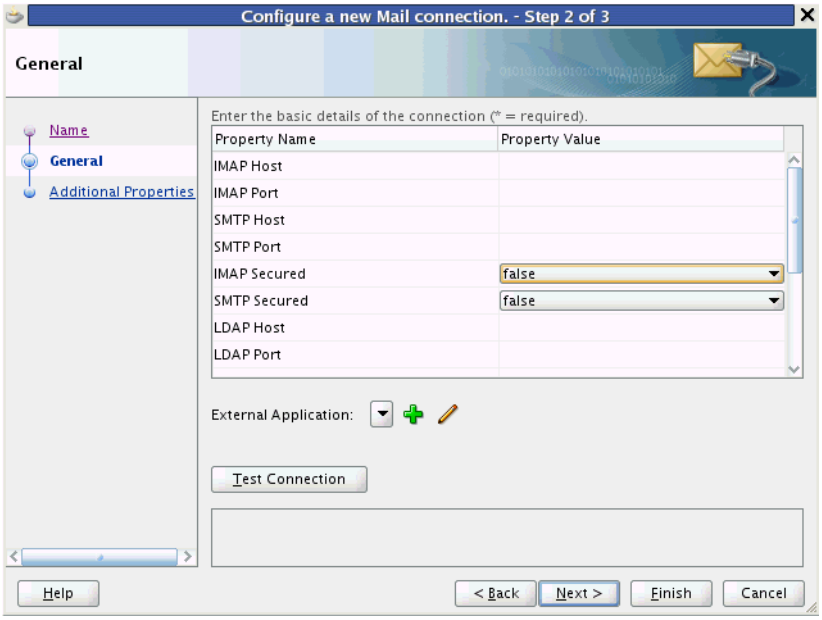
To use an external application definition with a secured service (such as a mail server or portlet producer) you associate the named external application with the connection configuration to the required service. For example, the connection to a mail server requires the user to supply a valid username and password to see their mail. Therefore, by associating an external application to the IMAP server connection definition, the user's credentials are automatically passed as part of the mail request as shown in Figure 69–12.

Note: The following WebCenter Portal services must be configured with external applications for credential mapping support to be available:

- Instant Messaging and Presence (IMP)
- Mail
- Documents
- RSS

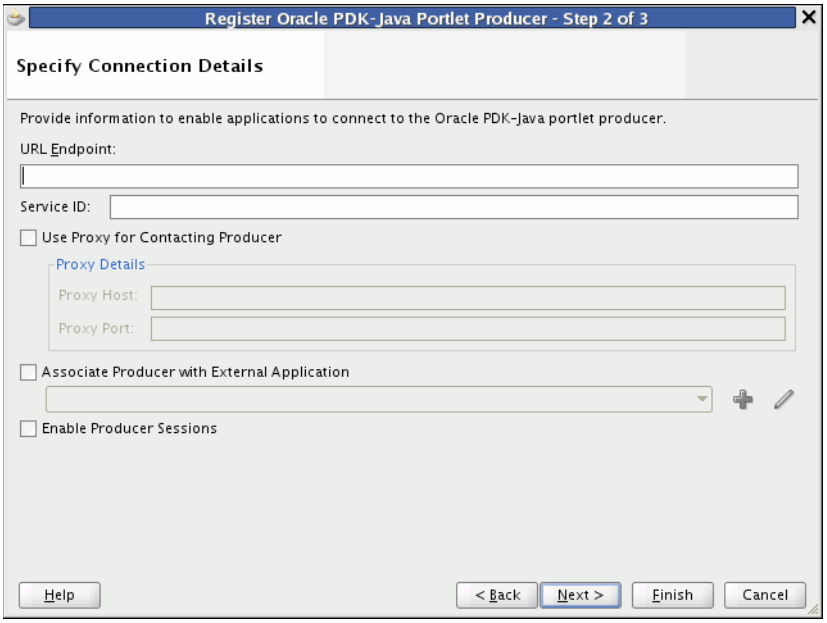
For more information about the identity propagation mechanisms used by WebCenter Portal services, see Section 69.16, "Identity Propagation Mechanisms".

Figure 69–12 Configure a New Mail Connection Page



When a portlet producer depends on an application that handles its own authentication, you can associate the producer with an external application so that when you register the producer it is a simple task to select the appropriate external definition that maps to the application that is exposed within the portlet, as shown in Figure 69–13.

Figure 69–13 External Application URL



At run time, the producer uses the information associated with the external application to authenticate the user to the application, and consequently consume its portlets. The producer code is responsible for actually performing the authentication interaction with the external application. The external application support provided

with WebCenter Framework simply provides the information needed for authentication to the portlet producer. The use of external applications is supported for both Oracle PDK-Java as well as WSRP producers.

For example, a producer provides a stock portfolio portlet from a portlet-producing application that has its own authentication mechanism. In this case the developer:

- Defines the external application. This can be done through the Oracle JDeveloper wizard or through Oracle Enterprise Manager.
- Associates the external application with the portlet producer.

69.13.1.2 Automated Single Sign-On

With automated single sign-on, the user directly links to the application and is automatically authenticated to the secured web application, as their credentials are retrieved from the credential store. This provides the end user with a seamless single sign-on experience.

Note: Automated login is not supported for:

- External applications using BASIC authentication.
- External applications configured for SSO.
- External applications with a customized login form (built using ADF Faces) that does not implement the J2EE security container login method `j_security_check` for authentication.
- External sites that do not support UTF8 encoding

Rather than using a URL directly to the web application, links to the application are proxied through the external application's automated login servlet (`adfextapplogin`). If the user is not authenticated to the external application and they have not previously stored their credentials in the credential store, they will be challenged for their password through the credential provisioning page discussed below. If, however, the user has previously defined credentials, they will be returned from the credential store and the user will automatically be logged on to the application.

The proxy URL references the external application in question and redirects to the URL that is specified in the external application definition.

```
/adfextapplogin?extappid=<extappid>
```

For example, if you had a WebCenter Portal: Framework application in which you defined an external application that represented the myoracle.com Web site (external application identifier is "myoracle"), the proxy URL would look like this:

```
/adfextapplogin?extappid=myoracle
```

The link's `target` attribute should also be set appropriately. For example, if you use `<a href=>`, then set the `target` attribute appropriately in addition to specifying the target in the `href`. The target attribute specified for the servlet will determine how the Cancel button functions as described below.

```
/adfextapplogin?extappid=<extappid> [target= _self | _blank]
```

If you specify `target=_blank` the link opens in new window. If you specify `target=_self` the link opens in current window. If the `target` parameter is not specified the link opens in current window.

This parameter also affects how the **Cancel** button on the credential provisioning page works. If `_blank` is specified, the new window is closed when **Cancel** is clicked; if `_self` is specified (or the `target` parameter is not used), the user is returned to the calling page.

Note: Automated login for external sites is not supported for sites that do not support UTF8 encoding.

69.13.2 Supplying User Credentials

How you allow an end user to define their credentials for an external application depends on the use of the external application. For most services, the credential provisioning screen is incorporated into the task flow that the service exposes and for these no further configuration steps are required. You can, however, add the `External Application - Change Password` task flow component to applications using these services thereby allowing the end user to preemptively set the appropriate user name and password for each of the external applications that is registered with your WebCenter Portal: Framework application.

The `External Application - Change Password` task flow displays all external applications defined in the application that do not specify shared credentials (for more information about shared credentials, see Section 69.13.3, "Managing External Applications"). Note that the user must to be authenticated to view this task flow.

For the Instant Messaging and Presence service, however, you must explicitly add the `External Application - Change Password` task flow component to your application from the Resource Palette or Component Palette. For step-by-step instructions on how to configure security for the Instant Messaging and Presence service using the `External Application - Change Password` task flow component, see Section 35.2.2, "Adding the IMP Service at Design Time".

At run time, the credential provisioning screen displays login data fields composed of the data fields specified through external application registration. Users fill in the data fields with their login information for the specified external application and that login information is passed to the external application or service. Entering the credentials in the provisioning screen also results in the credentials being persisted in the credential store configured for the WebLogic domain.

By default, the login information the user entered is preserved in a credential store, which handles logins for future sessions. The user does not have to enter login information again (unless the user's credentials change). However, the end user can choose to use the information for the current session by deselecting the **Remember my Login Information** checkbox on the credential provisioning page.

69.13.3 Managing External Applications

This section provides information about registering external applications. Additionally, it describes the process of editing and deleting registration details. It contains the following subsections:

- Section 69.13.3.1, "Adding External Application Task Flows"
- Section 69.13.3.2, "Working with External Applications in Oracle JDeveloper"
- Section 69.13.3.3, "Working with External Applications in Enterprise Manager"
- Section 69.13.3.4, "Working with External Applications Using WLST"

69.13.3.1 Adding External Application Task Flows

The External Application task flow enables users to manage external application connections at runtime. This task flow provides an interface for users to update their credentials for all external applications as and when these credentials are changed at the back end. That is, users do not need to update credentials through the runtime Administration Console's Services tab if they use this task flow.

The external application task flows that you can add using Oracle JDeveloper are:

- **External Application**: This task flow enables users to register, modify, and delete connections at runtime.
- **External Application - Change Password**: This task flow enables users to change external application passwords at runtime.

For applications created using the Framework application template, both these task flows are available out-of-the-box through Framework application Administration page (Services tab). For details, see the section "Managing Services, Portlet Producers, and External Applications" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

In addition, just like other task flows, you can add external application task flows to your application pages. This may be especially useful if you are not using the Framework application template and the Administration pages are therefore not part of your project.

Special permissions are required to manage or view external application connections through these task flows:

- **AppConnectionManager** - Users with this role can register, modify, and delete external application connections at runtime.
- **AppConnectionViewer** - Users with this role can view external application connections at runtime. By default, any user who is logged in (that is, has the authenticated-role) is granted this role.

By default, users with the Administrator role can manage external applications. If you want other users to manage connection through these task flows you must grant them the AppConnectionManager role.

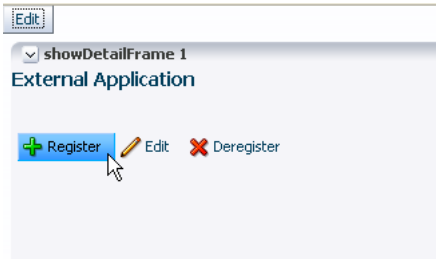
To add the external application task flows:

1. Create or open a JSF page in your application where you want the task flow to be added (see Section 5.3, "Adding Pages to a Portal").
2. In the Resource Palette, expand **My Catalogs**, **WebCenter Portal - Services Catalog**, and **Task Flows**.
3. Drag **External Application** from the Resource Palette and drop it onto the page inside of the `af:form` `begin` and `end` tags.
4. Drag **External Application - Change Password** from the Resource Palette and drop it onto the page inside of the `af:form` `begin` and `end` tags after the **External Application** task flow (`af:region - #{bingings.extapp1.regionModel}`).
5. Grant the AppConnectionManager role to one or more test users, if required:
   - Add the test user `TEST_EXTAPP`.
   - Grant the AppConnectionManager role.

For information about how to add a user and grant this role, see the section "Creating Test Users" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6. Save and run your page. Login as an administrator or the `TEST_EXTAPP` user defined in the previous step. The screen shown in Figure 69–14 appears.

Figure 69–14 External Application Task Flow in a Browser



Note: At runtime, application administrators can grant users the AppConnectionManager and AppConnectionViewer roles through WebCenter Portal administration (see the section "Adding Members to Application Roles" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.). Alternatively, system administrators can grant AppConnectionManager and AppConnectionViewer roles through Fusion Middleware Control (see, "Granting Application Roles Using Fusion Middleware Control" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.)

69.13.3.2 Working with External Applications in Oracle JDeveloper

This section provides information about registering external applications and editing and deleting registration details in Oracle JDeveloper. It contains the following subsections:

- Section 69.13.3.2.1, "Registering an External Application in Oracle JDeveloper"
- Section 69.13.3.2.2, "Editing External Application Registration Details in Oracle JDeveloper"
- Section 69.13.3.2.3, "Deleting External Application Registration Details in Oracle JDeveloper"

69.13.3.2.1 Registering an External Application in Oracle JDeveloper

Use the Register External Application Wizard to identify and store information about the type of data required to authenticate to an external application, such as the names of login fields.

To register an external application in Oracle JDeveloper:

1. In the Applications Navigator, right-click a WebCenter Portal: Framework application or project and select **New** from the context menu.
2. In the New Gallery, select **External Applications** under the **General** node.
3. In the right pane, select **External Application**, and click **OK**.

   This displays the Register External Application Wizard.

4. On the Name page, use the **Create external application in** option to specify whether the external application can be reused in other Framework applications. Select **Application Resources** to make the external application available only in the Framework application in which it is registered, or select **Resource Palette** to make the external application available from the Resource Palette to any new Framework applications you create in Oracle JDeveloper.

   If you choose **Resource Palette**, then the external application connection will be visible under IDE connections in the Resource Palette. If you want to use it in an application, you can right click the external application from resource palette and click **Add to Application**.

5. In the **Name** field enter a unique name to identify the application.

   This name must be unique within the WebCenter Portal: Framework application, and among other connections as well. Note that you cannot edit this field afterward.

6. In the **Display Name** field enter a name for the application that end users will see in the credential provisioning screens.

   You can modify the Display Name at any time as described in Section 69.13.3.2.2, "Editing External Application Registration Details in Oracle JDeveloper". If you leave the field blank the display name is set to the value of the Name field by default.

7. Click **Next**.

8. On the General page, in the **Login URL** field enter the URL to which the HTML login page is submitted.

   View the HTML source of the application's login form to retrieve this URL.

   > **Note:** The fields on the General pane are only required if the external application being defined participates in click-through login as described in Section 69.13.2, "Supplying User Credentials".

9. In the **User Name/ID FieldName** field, enter the label that the application uses for the user name field, for example, `User Name`.

10. In the **Password FieldName** field, enter the label that the application uses for the password field, for example `Password`.

11. From the **Authentication Method** list, select the application's login method.

    Choose from the following:

    - **GET**

      Presents a page request to a server. Submits the login credentials as part of the login URL.

    - **POST**

      Submits login credentials within the body of a form.

12. Click **Next**.

13. On the Additional Fields page, enter the names and values of any additional fields that are submitted with the external application's login form:

    Click the **Add Field** button to create an input field:

- **Field Name**

  Enter a unique name for any additional field that requires user input on the external application HTML login form.

- **Field Value**

  Enter a default value for the corresponding field name.

- **Display to User**

  Select to display the field on the external application login screen. If the field is not displayed (unchecked), then a default value must be specified, which will be used to login into the external application for all users. If the value is user-specific, then the field must be displayed to the user provisioning page.

  Note: The Delete Field option can be used to delete selected rows.

14. Click **Next**.
15. On the Shared Credentials page, select the **Specify Shared Credentials** check box and specify a **Username** and **Password** if you want all authenticated users to access the external application using this credential. Authenticated users will not be challenged to provide their credentials when they access the external application.
16. Click **Next**.
17. On the Public Credentials page, select the **Specify Public Credentials** check box and specify a **Username** and **Password** to be used for all unauthenticated (public) users accessing the external application. This is required when the external application content is accessed through one of the WebCenter Portal services (such as Document Library, or Instant Messaging and Presence), and the task flow is placed on a public page.
18. Click **Finish** to register the external application.

After registering your external application, you must configure the application to allow the end user to define the username and password. You can do this by dropping an `External Application - Change Password` task flow component into your application. This allows the end user to preemptively set the appropriate username and password for each of the external applications that is registered with your WebCenter Portal: Framework application.

To configure the application to allow the end user to define the username and password, perform the following steps:

1. Open a JSF Page in your ViewController project.
2. From the Resource Palette, under My Catalogs, WebCenter Portal - Services Catalog, expand **Task Flows**.
3. Drag an `External Application - Change Password` task flow component onto your page.
4. When prompted, choose **Region** as the way to create the task flow.
5. Secure the page as this task flow is accessible only to authenticated users

69.13.3.2.2 Editing External Application Registration Details in Oracle JDeveloper

Use the Edit External Application Registration Information Wizard to revise the registration details provided for an external application.

To edit external application registration details:

1. In the Application Navigator, from the Application Resources pane under the Connections node, right-click an external application and select **Properties** from the context menu.
2. In the Edit External Application Wizard, click a link to show a page and revise its values.

   Choose from the following:

   - **Name**
   - **General**
   - **Additional Fields**
   - **Shared Credentials**
   - **Public Credentials**

   For more information, see Section 69.13.3.2.1, "Registering an External Application in Oracle JDeveloper".
3. Click **OK** to save your changes and exit the wizard, or click **Cancel** to exit the wizard without saving.

69.13.3.2.3 Deleting External Application Registration Details in Oracle JDeveloper To delete external application registration information, perform the following steps:

1. In the Application Navigator, from the Application Resources pane under the Connections node, right-click an external application and select **Delete** from the context menu.

   Alternatively, you can select an external application in the Applications Navigator and from the Edit menu, select **Delete**.
2. In the External Application Delete dialog box, select **Yes**.

Oracle recommends that you remove any references to services, such as a portlet producer, with which the external application is associated. Failing to do so will likely result in runtime errors, because the components will attempt to communicate with the external application.

69.13.3.3 Working with External Applications in Enterprise Manager

Just as you can create, edit, and delete external application registration details in Oracle JDeveloper, you can also do this in Enterprise Manager. See the section "Working with External Applications" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

69.13.3.4 Working with External Applications Using WLST

As well as Oracle JDeveloper and Enterprise Manager, you can also use WebLogic Scripting Tool (WLST) commands to create, edit, and delete external application registration details. See the section "Working with External Applications" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

69.14 Registering Custom Certificates with the Keystore

Secure Sockets Layer (SSL) Communication requires the use of trusted certificates issued by a certificate authority, which vouches for the authenticity of the certificates that it issues or signs. Widely accepted certificate authorities are listed in the keystore,

the `cacerts` file, available in the `<JDEV_HOME>\jdk\jre\lib\security` directory. If a portlet producer uses a security certificate issued by a non-widely accepted certificate authority and you try to access portlets from this producer, a security alert is displayed informing you that the security certificate was issued from a certificate authority you do not trust. This means the certificate is not available in the keystore. To avoid being prompted each time you access such portlets, you must register this certificate with the keystore.

To register a certificate with the keystore, perform the following steps:

1. Navigate to *JDEV_HOME*`\jdk\jre\lib\security`.
2. Back up the `cacerts` file.
3. Access the producer URL in Internet Explorer to get the certificate.

> **Note:** Recent versions of FireFox do not provide a means to export certificates.

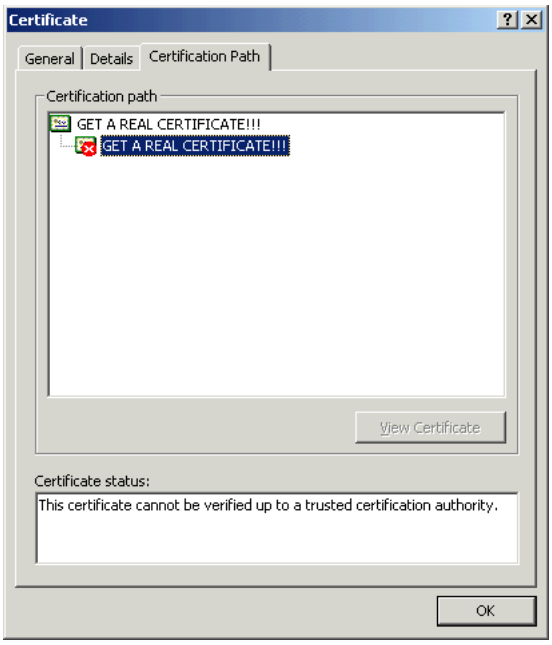
4. In the Security Alert dialog box, shown in Figure 69–15, click **View Certificate**.

Figure 69–15 Security Alert Dialog Box



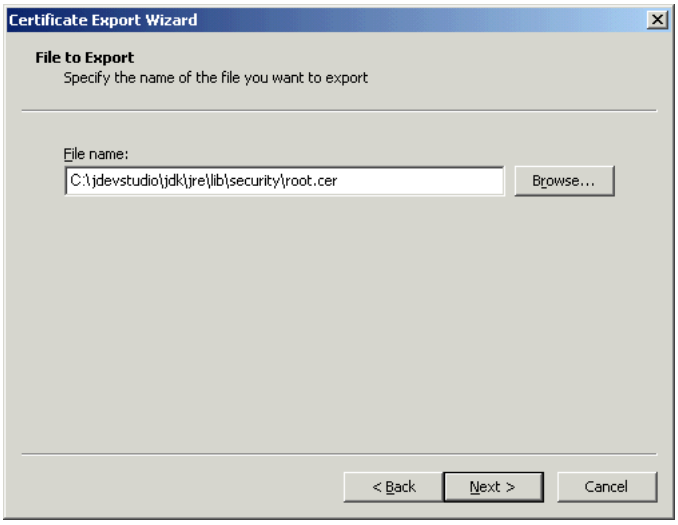
5. In the Certificate dialog box, click the **Certification Path** tab.
6. The dummy child certificate is selected by default as shown in Figure 69–16. Select the root certificate and click **View Certificate**.

Figure 69–16 Certificate Dialog Box



7. Click the **Details** tab, and click **Copy to File**.
8. In the Certificate Export Wizard, accept the default settings and click **Next** until you reach the File to Export screen, shown in Figure 69–17.

Figure 69–17 File to Export Screen of the Certificate Export Wizard



9. In the File Name field, enter `<JDEV_HOME>\jdk\jre\lib\security\root.cer` and click **Next**.
10. Click **Finish**.
11. In the command prompt, set your default directory to `<JDEV_HOME>\jdk\jre\lib\security` and run the following command:

    ```
    keytool -import -file root.cer -keystore cacerts -storepass changeit
    ```

    By running this command, the `root.cer` certificate is imported into the keystore.

12. Enter `y` at the prompt to confirm that you trust this certificate.
13. Verify that the `cacerts` file is updated with the certificate.

69.15 Overriding Inherited Security on Portlets and Customizable Components

Individual actions on portlets and customizable components are not secured by default. Rather, the ability to customize a portlet or customizable component as a whole is inherited from the page permissions. If you want to grant more granular activities within a portlet or customizable component, then you can override the page-level security inheritance and define security directly on the required actions.

The ability of a user to perform actions on portlets and customizable components is inherited from the page security based on the value of the application-wide switch, `enableSecurity`, in the `adf-config.xml` file. If you selected the WebCenter Portal: Framework application template while creating your application, then the `adf-config.xml` file is located in the `<APPLICATION_NAME>/.adf/META-INF` directory. The `enableSecurity` element is not available by default in `adf-config.xml`. To override or extend the page-level security inheritance for portlets and customizable components, you must add the portlets security and customizable components security sections in the `adf-config.xml` file, as shown in Example 69–1 and Example 69–2 and set the `enableSecurity` element in those sections to `true`.

Example 69–1 enableSecurity Element in the Portlet Security Section in adf-config.xml

```
<!--
================================================================================
PORTLETS ACTIONS SECURITY
================================================================================
-->
<adfp:adf-config-child xmlns:adfp="http://xmlns.oracle.com/adfp/portlet">
  <adfp:enableSecurity value="true"/>
    <adfp:actionsCategory>
       ......................................
</adfp:adf-config-child>
```

Example 69–2 enableSecurity Element in the Customizable Components Security Section in adf-config.xml

```
<!--
================================================================================
CUSTOMIZABLE COMPONENTS ACTIONS SECURITY
================================================================================
-->
<cust:customizableComponentsSecurity
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable/config">
  <cust:enableSecurity value="true"/>
    <cust:actionsCategory>
       ......................................
</cust:customizableComponentsSecurity>
```

Security for actions on portlets and customizable components can be implemented at the following levels:

- Page level: You can define security for portlets and customizable components such that page-level privileges are inherited by these components. This is the default behavior.

By default, portlets and customizable components inherit allowable actions from the defined page-level permissions such as personalize or customize. That is, a user who has *customize* privileges on the page has permission on the customize action for the components on that page. The `enableSecurity` element enables you to override the security inheritance behavior and can take either of the following values:

- `true`: If set to `true` (the default), then the ability for a user to modify a component will first be determined from the page permissions and then adjusted according to the current set of actions defined for that type of permission. If a user has customize permission, then the actions that constitute the customize category (move, customize, and so on) are available to the user, but they will be overridden by the actions that are defined in the `adf-config.xml` file. For example, a page designer wants to allow the end user to be able to customize portlets, but not customize the page layout. By setting `enableSecurity` to `true`, the page designer enforces that users must first have customize permission on the page. Setting `customizeActionsCategory` to `false` for customizable components will prevent the customization of the page layout, yet still allowing portlet customization. (As the default for `customizeActionsCategory` is `true`, it does not need to be set explicitly for portlets.).
- `false`: If set to `false`, then all the actions are available to users. The user's page permissions and actions configured in `adf-config.xml` are ignored.

- Actions category level: You can define security on all actions for portlets or customizable components that belong to a named category.

  You can add an `actionsCategory` element in the `adf-config.xml` file to define security on multiple actions simultaneously. Depending on the `actionCategory` attributes that you enable, appropriate privileges are provided on the portlets or customizable components.

- Actions level: You can define security on individual actions for portlets or customizable components.

  You can use the `actions` element in the `adf-config.xml` file to enable or disable individual actions. Depending on the `actions` attributes that you enable, appropriate privileges are provided on the portlets or customizable components.

Notes:

- Privileges can be inherited from the parent only. Inheritance from a component in any other position in the hierarchy is not supported
- Although the security override implementation for portlets and customizable components is similar, they are independent from each other. Therefore, if you place a portlet inside a customizable component (for example, in a `PanelCustomizable` component), the portlet will not inherit override settings from the customizable component. Instead it will use the security override settings that are defined for portlets.
- Settings made at the actions category level or actions level are applicable for all component instances in the application. These settings cannot be made for a single instance of a portlet or customizable component.

Section 69.15.1, "Portlets Security" describes how you can implement security on portlets at the actions category level and actions level. For information on how to define security for actions on customizable components at the application level, see Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."

69.15.1 Portlets Security

You can define portlet security if actions on portlets are inherited from the page at the application level by setting `enableSecurity` to `true` in the portlets security section of the `adf-config.xml` file. A value of `true` implies that the user's permissions are determined from the page permission and then augmented according to the `actionsCategory` and `actions` elements specified. By defining actions categories and individual actions, you can control the exposure of the individual actions available within the given page permissions.

To implement security for actions on portlets at various levels as described earlier, you must define security settings at the following sections:

- Defining Security at the Actions Category Level
- Defining Security at the Actions Level

69.15.1.1 Defining Security at the Actions Category Level

You can add an `actionsCategory` element in the portlets security section in the `adf-config.xml` file to define the group of actions that are exposed on the portlets within the application. Depending on the `actionsCategory` attributes that you enable, appropriate privileges are provided on the portlets. Table 69–3 describes the different `actionsCategory` attributes and the portlet actions they support by default.

Table 69–3 actionsCategory Attributes and Portlets Actions Mapping

Attribute Value	Actions Supported
viewActionsCategory	Render isHelpModeAvailable isNormalModeAvailable isAboutModeAvailable isPreviewModeAvailable isDetailModeAvailable isLinkModeAvailable isPrintModeAvailable
customizeActionsCategory	showMoveAction showRemoveAction isCustomizeModeAvailable showMinimizeAction showMaximizeAction isConfigModeAvailable
personalizeActionsCategory	isPersonalizeModeAvailable

Example 69–3 shows the `actionsCategory` entry that you can add to the portlets security section in the `adf-config.xml` file. In this example, `customizeActionsCategory` is set to `false` to prevent customization. You can use Expression Language (EL) for the values of these elements.

Example 69–3 actionsCategory Element in the Portlets Security Section

```
<!--
================================================================================
PORTLETS ACTIONS SECURITY
================================================================================
-->
<adfp:adf-config-child xmlns:adfp="http://xmlns.oracle.com/adfp/portlet">
  <adfp:enableSecurity value="true"/>
    <adfp:actionsCategory>
      <adfp:actionCategory name="viewActionsCategory" value="true"/>
      <adfp:actionCategory name="customizeActionsCategory" value="false"/>
      <adfp:actionCategory name="personalizeActionsCategory" value="true"/>
    </adfp:actionsCategory>

    <adfp:actions>
    ......................................
    </adfp:actions>

</adfp:adf-config-child>
```

69.15.1.2 Defining Security at the Actions Level

You can use the `actions` element in the portlets security section of the `adf-config.xml` file to enable or disable individual portlet actions. Depending on the `action` attributes that you enable, appropriate privileges are provided on the portlets.

Example 69–4 shows an example of an `actions` entry that you can add to the portlets security section in the `adf-config.xml` file. You can use EL for the values of these elements. In this case you prevent customization by setting `isCustomizeModeAvailable` to `false`.

Example 69–4 actions Element in the Portlets Security Section

```
<!--
================================================================================
PORTLETS ACTIONS SECURITY
================================================================================
-->
<adfp:adf-config-child xmlns:adfp="http://xmlns.oracle.com/adfp/portlet">
  <adfp:enableSecurity value="true"/>
    <adfp:actionsCategory>
    ......................................
    </adfp:actionsCategory>

    <adfp:actions>
      <adfp:action name="Render" value="true"/>
      <adfp:action name="showMoveAction" value="true"/>
      <adfp:action name="isCustomizeModeAvailable" value="false"/>
      <adfp:action name="isPersonalizeModeAvailable" value="true"/>
    </adfp:actions>

</adfp:adf-config-child>
```

Using EL to Prevent Customization of Portlets Outside of Business Hours

An example to show when you may need to override inherited portlet security is an application that is configured to disable portlet customization outside of standard business hours. For this, you must first create a managed bean (for example, a managed bean called `appBusinessRules`), containing the method shown in Example 69–5.

Example 69–5 InsideBizHours Method Defined in appBusinessRules Managed Bean

```
public boolean isInsideBizHours()
  {
    Calendar rightNow = Calendar.getInstance();
    int      currentHr = rightNow.get(rightNow.HOUR_OF_DAY);

    // Do not allow customize operation outside of standard business hours

    if ((currentHr > 9) && (currentHr < 17))
       return true;
    else
       return false;
  }
```

You can then reference this managed bean from the `actionsCategory` element in the portlet security section of the `adf-config.xml` file, as shown in Example 69–6.

Example 69–6 InsideBizHours Method Referenced in the adf-config.xml File

```
<adfp:adf-config-child xmlns:adfp="http://xmlns.oracle.com/adfp/portlet">
  <adfp:enableSecurity value="true"/>

    <adfp:actionsCategory>
      <adfp:actionCategory name="customizeActionsCategory"
      value="#{appBusinessRules.InsideBizHours?"true":"false"}"/>
    </adfp:actionsCategory>

</adfp:adf-config-child>
```

In this example, the `customizeActionsCategory` will be set to `true` only if the application is run within business hours. Outside of these hours, the portlet cannot be customized even if the user had that permission granted on the page. All other categories that are not explicitly defined, will be inherited from the page.

Using EL to Prevent Personalization and Customization of Portlets Outside the Corporate Network

In this example the managed bean checks the IP address of the request to determine whether the user has accessed the application through the corporate proxy server or from within the corporate network. In this simple example, assume that if the request has the proxy server's IP address, then it is coming from outside the corporate network. In general it is not advised to base security strictly on IP addresses, because these can be compromised. For this, you must add the method shown in Example 69–7 to the managed bean:

Example 69–7 InsideCorpNetwork Method Defined in appBusinessRules Managed Bean

```
public boolean isInsideCorpNetwork()
  {
    // Do not allow personalize and customize operations
    // for requests that go through the corporate proxy
```

```
    FacesContext       ctx = FacesContext.getCurrentInstance();
    ExternalContext    ectx = ctx.getExternalContext();
    HttpServletRequest myrequest = (HttpServletRequest) ectx.getRequest();
    String             currentIP = myrequest.getRemoteAddr();

    if (currentIP.equals(getProxyServerIP()))
        return false;
    else
        return true;
  }
```

You can then reference this managed bean from the `actionsCategory` element in the portlet security section of the `adf-config.xml` file, as shown in Example 69–8.

Example 69–8 InsideCorpNetwork Method Referenced in the adf-config.xml File

```
<adfp:adf-config-child xmlns="http://xmlns.oracle.com/adfp/portlet">
  <adfp:enableSecurity value="true"/>

    <adfp:actionsCategory>
      <adfp:actionCategory name="customizeActionsCategory"
                    value="#{appBusinessRules.InsideCorpNetwork?"true":"false"}"/>
      <adfp:actionCategory name="personalizeActionsCategory"
                    value="#{appBusinessRules.InsideCorpNetwork?"true":"false"}"/>
    </adfp:actionsCategory>
</adfp:adf-config-child>
```

In this example, the `customizeActionsCategory` and the `personalizeActionsCategory` will be set to `true` only if the IP address of the request for the application does not match that of the corporate proxy. The assumption is that the internal requests would have a valid client IP address. All other categories that are not explicitly defined, will be inherited from the page.

69.16 Identity Propagation Mechanisms

The following table lists the identity propagation mechanisms employed by WebCenter Services for propagating the end-user's identity to the various information sources from which content is being integrated into the Framework application.

Whenever possible, WS-Security is the preferred means of identity propagation. Where WS-Security cannot be used due to legacy restrictions or pre-defined store-specific standards or specifications, the primary mechanism used is the credential mapping capability provided by the External Applications service to obtain the user's credentials for a remote application using a distinct security provider. For more information about WS-Security, see the *Oracle Application Server Web Services Security Guide*. For more information about External Applications, see Section 69.13, "Working with External Applications".

Table 69–4 Identity Propagation Mechanisms

Service	Connection Type	Identity Propagation Mechanism
Discussions	Jive	Oracle WS-Security
Announcements	see Discussion Forum	

Table 69–4 (Cont.) Identity Propagation Mechanisms

Service	Connection Type	Identity Propagation Mechanism
Documents	Content Server	Proprietary ID propagation mechanism through socket connection. Can use SSL with mutual authentication, or clear socket with IP authorization (or use External Application option)
	File System	N/A
	Portal 10g/11g	JSR-170 (External Application)
	Day Adapters	JSR-170 (External Application)
EMail	EMail Connection	External Application
	LDAP Connection	External Application
Events - Personal	Exchange Server Connection	External Application
External Application	HTTP Request from Browser	External Application
IMP	Microsoft Live Communication Server (LCS)	SOAP, Web Services calls. External Application
Portlet Producers	WSRP Producer	Oracle WS-Security
	(Secure)	(Recommended by WSRP Specification)
	WSRP Producer	WSRP userContext in SOAP message
	(Non-Secure)	(WSRP Specification)
	JPDK Producer	External Application
	(External App)	JPDK payload includes the user information and is conveyed to the producer in the ProviderUser. The information also includes a mapped username and password. (Proprietary, Legacy)
	JPDK Producer	Username in render request
	(Non-Secure)	(Proprietary, Legacy)
Search	Secure Enterprise Search (SES)	Web Service Call
Wiki	Browser Connection	SSO mechanisms - OSSO/OAM or other WLS supported SSO mechanism
Worklist Service	BPEL Connection	Web Service call Oracle WS-Security

69.17 Securing Identity Propagation Through WSRP Producers with WS-Security

The Web Services for Remote Portlets (WSRP) specification indicates that Web Services Security (WS-Security) can be leveraged for providing secure identity propagation between the consumer and the portlet producer. However, WSRP in and of itself does not provide secure identity propagation of the end user's identity to the portlet producer. The WSRP specification explicitly defers to other security standards for secure identity propagation and does not go into the specific WS-Security profiles or

options that may be employed. In the absence of a secure mechanism, WSRP defines the concept of *user categories*, which can be mapped to security roles like the ones used by the JSR168 portlets. By using a combination of WSRP and WS-Security, you can ensure end-to-end security.

This section covers the following:

- Identity Propagation Without WS-Security
- Identity Propagation with WS-Security
- Configuring Security for WSRP Portlets

69.17.1 Identity Propagation Without WS-Security

When using WSRP without WS-Security, the `userContext` structure within the SOAP message contains user profile information and user category information. This information is not considered secure and should only be used for personalization and customization functionality. It should not be used for authorization of sensitive resources. This information is also exposed in the JSR168 APIs, `isUserInRole(`*`role`*`)` and `getUserPrincipal`. The code in Example 69–9 shows how a sample portlet's markup rendering code uses the `isUserInRole` API to decide what content to display.

Example 69–9 isUserInRole(role) API

```
private void doViewHtml(RenderRequest
request, RenderResponse response)
throws PortletException, IOException
{
// To do: markup the required content.
PrintWriter out = response.getWriter();
out.print("<p>Welcome");
out.println("</p>");
if (request.isUserInRole("moderator"))
   {out.println("<p>MODERATOR</p>" );}
else
   {out.println("<p>not moderator</p>" );}
if (request.isUserInRole("participant"))
   {out.println("<p>PARTICIPANT</p>" );}
else
   {out.println("<p>not participant</p>" );}
if (request.isUserInRole("viewer"))
   {out.println("<p>VIEWER</p>" );}
else
   {out.println("<p>not viewer</p>" );}
    }
```

69.17.2 Identity Propagation with WS-Security

When WS-Security is leveraged with WSRP, the user's identity is propagated outside of the SOAP message body in the WS-Security header. This is a user assertion, using the Username Token format, and is digitally signed to authenticate the consumer and to ensure the integrity of the assertion.

When this mechanism is used, the JSR 168 APIs `isUserInRole` and `getUserPrincipal` are established based on the security context resulting from the WS-Security authentication, rather than the information in the SOAP message's `userContext`.

Also, when an Anonymous (i.e., PUBLIC) client is mapped to a default user using the producer's Default User connection parameter, the WSRP producer must be configured with `strict-authentication` and a grant must be added to the Policy store. For more information, see "Adding a Grant to the Policy Store for a Mapped User Identity" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The use of WS-Security adds some complexity to the configuration and management of the WebCenter Portal: Framework application and the set of producers it consumes. However, when the situation warrants its use, it becomes an important ingredient of the SOA architecture that ensures the security of the information being published by the WebCenter Portal: Framework application.

Oracle WebCenter Framework supports the following token profiles (to digitally sign the security token and message body to ensure authenticity and integrity):

- Username token without password
- Username token with password
- SAML token (uses the *sender vouches* method that the producer uses to confirm the subject assertion)

Digitally signing the security token and the SOAP message body accomplishes the following objectives:

- Consumer Authentication
- Assertion and Message Integrity
- Supported Producers
- Security Domain Implication

Consumer Authentication

When a portlet producer is generating sensitive information, for example paystub information, it is imperative that it only responds to requests to show the information from a legitimate consumer.

By using WS-Security, and having the consumer digitally sign the security token and the message body, the producer can verify the signature using the public key of the legitimate consumer. If the signature cannot be verified, then it means that the request may have come from a fraudulent consumer. By requiring the verification of the digital signature, the sensitive information will only be sent to the legitimate consumer.

Assertion and Message Integrity

In addition to verifying the identity of the consumer making the Web Service requests, digitally signing the security token and the message body also ensures that the token and the message have not been tampered with. This prevents such problems as man-in-the-middle attacks where a legitimate request might be intercepted and the user name in the security token replaced with another user name to see the paystub information coming back for the other user. By digitally signing the token, it cannot be tampered with. Any modification to the token would result in the inability to verify the signature on the producer end, and would result in a SOAP fault to be returned instead of the requested paystub information.

Supported Producers

WS-Security implementation is supported by the Oracle WebCenter Suite WSRP container. Other WSRP vendors may also be able to support the WS-Security

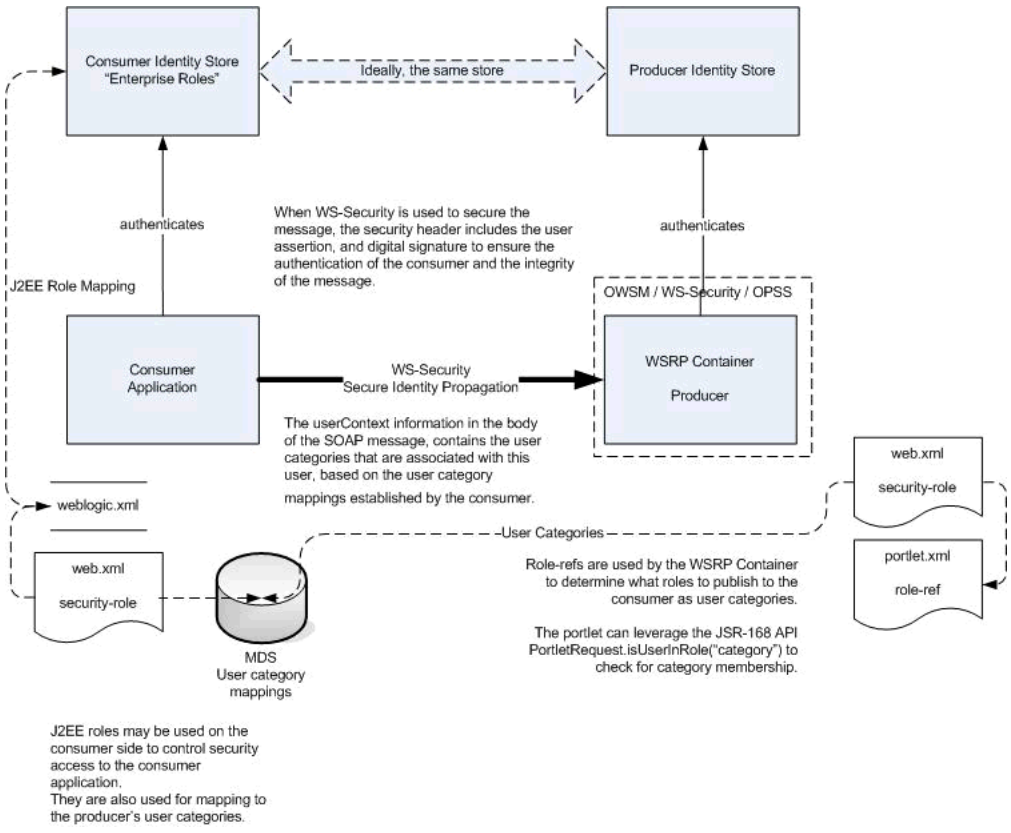
configuration of Username Token without password, with XML digital signature on the Username Token and the SOAP Message body.

Security Domain Implication

When using secure identity propagation as described in this section, the user name of the user authenticated to the consumer (WebCenter Portal: Framework application) is propagated to the producer without any remapping or providing any credentials. There is an inherent assumption that the producer understands this user name and can locate this user in its associated security domain. Consequently, it is highly desirable to ensure that the consumer and producer share the same security provider (identity store) to simplify the management of this configuration.

Figure 69–18 summarizes the overall WSRP portlet security architecture.

Figure 69–18 WSRP Portlet Security Architecture



69.17.3 Configuring Security for WSRP Portlets

Before you configure the producer for WS-Security, you must first deploy your standards-compliant portlet producer to the Oracle WebCenter Suite WSRP Container by performing the steps described in Chapter 62, "Testing and Deploying Your Portlets".

After you have deployed the producer, configure the producer for WS-Security by performing the following steps:

- Attaching a policy to the producer endpoint
- Creating the keystores
- Configuring the producer

- Configuring the consumer

These steps are described in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal* .

69.18 Implementing PDK-Java Portlet Security

This section describes the available security services for your PDK-Java portlet.



For more detailed information about the PDK classes referred to in this section, see the JavaDoc on OTN by clicking **Java Doc API** on the Portlet Development page available at

http://www.oracle.com/technology/products/ias/portal/portlet_development_10g1014.html

69.18.1 Assumptions

The following assumptions are made to perform the tasks in this section:

1. You have followed through and understood Section 60.2, "Creating Java Portlets".
2. You built a portlet using the wizard and successfully added it to a page.

69.18.2 Introduction to PDK-Java Portlet Security Features

This section introduces the major features that are available to secure your PDK-Java portlet producers.

69.18.2.1 Identity Propagation

When a user first logs in, they must enter their password to verify their identity and obtain access.

Once the user is authenticated, the producer code has access to the authenticated user's identity from the `PortletRenderRequest` that is available from the `HttpServletRequest` object as follows:

```
PortletRenderRequest pr = (PortletRenderRequest)request.getAttribute
   (HttpCommonConstants.PORTLET_RENDER_REQUEST);
   String userName = pr.getUser().getName();
```

When using this user identity for sensitive operations, it is important to ensure that you have configured this producer to use basic message authentication to secure the integrity of the identity assertion.

69.18.2.2 Authorization

Authorization determines if a particular user may view or interact with a portlet. For information about how to restrict access to portlets, see Section 69.18.4, "Portlet Security Managers."

69.18.2.3 Message-level Security

To this point, user authentication and authorization are covered, which do not check the authenticity of messages received by a producer. To completely secure your producers, you should also secure the communication with a producer. If the communication is not secured, then it is possible for someone to imitate an application instance and fool the producer into returning sensitive information. There are three types of communication security:

- **Server Authentication** restricts access to a producer to a small number of recognized computers. This method compares the IP address or the host name of an incoming HTTP message with a list of trusted hosts. If the IP address or host name is in the list, then the message is passed to the producer. If not, it is rejected before reaching the producer.
- **Message Authentication** provides consumer authentication and message integrity. It uses a shared key known to the client (Framework application) and the producer to digitally sign messages. See Section 69.18.5, "Message Authentication" for more information.
- **Message Encryption** relies on the use of the HTTPS protocol for communication between applications and producers. Messages are strongly encrypted to protect the data therein. Encryption provides a high level of security, but it incurs a performance penalty due to the additional processing required for each message.
- **User Input Escape** causes the application to escape any user input strings and treat them as text only to protect against XSS attacks, where an attacker attempts to pass in malicious scripts through user input forms. For example, if a portlet title is customizable, then an attacker might attempt to pass scripts or commands to the portlet through the title parameter entry. For this reason, the default behavior is to escape user input and thus disable any incoming scripts. See Section 69.18.6, "User Input Escape" for more information.

69.18.3 Single Sign-On

Portlets may act as windows into an application. They display summary information and provide a way to access the full functionality of the application. Portlets display some portions of the application in the WebCenter Portal: Framework application and typically enable the user to perform some application tasks.

An application may need to authenticate the user accessing the application through the portlet. The following are the possible application authentication methods:

- External Application. In this case, the Oracle Portal (Oracle Portal) user is different from the application user, but the application user name and password are managed by the Oracle Portal user.
- No Application Authentication. In this case, the communication between producer and consumer is not protected at all.

69.18.3.1 External Application

An external application uses a different authentication server than the WebCenter Portal: Framework application. This means that when a user is logged into the WebCenter Portal: Framework application, you want to also log them into the external application without having to type in their user name or password.

Applications that manage the authentication of users can be loosely integrated with the WebCenter Framework if the administrator registers them as external applications. When a user who was previously authenticated by the WebCenter Framework accesses an external application for the first time, WebCenter Framework attempts to authenticate the user with the external application. The authentication process submits an HTTP request that combines the registration information and the user's user name and password for the application. If the user has not yet registered their user name and password for the external application, then the user is prompted for the required information before making the authentication request. When a user supplies a user name and password for an external application, WebCenter Framework maps the new user name and password to the WebCenter Portal: Framework application user name

and stores them. They will be used the next time the user needs authentication with the external application.

The advantages of an external application implementation are as follows:

- Provides a single sign-on experience for users. However, users still must maintain different user names and passwords. In addition, the external application user name mapping must be maintained.
- Allows integration with multiple portals independent of their user repositories and Oracle Single Sign-On.
- Avoids the requirement of having access to the application source code.

The disadvantages of an external application implementation are as follows:

- Does not share the same user repository as the portal, which requires additional maintenance of user information by the end user.
- Transmits the user name and password to the producer in plain text, unless you implement Secure Sockets Layer (SSL).

69.18.3.2 No Application Authentication

The producer trusts the WebCenter Portal: Framework application instance sending the request completely. The producer can determine if the user is logged in and the portal user name, but the application has not authenticated the user.

The advantages of no application authentication are as follows:

- Provides the easiest form of integration and the fastest to implement.

The disadvantages of no application authentication are as follows:

- Provides the least security.
- Provides the weakest integration with the WebCenter Portal: Framework application.

69.18.4 Portlet Security Managers

Portlet security managers may be implemented within a producer to restrict access to portlets depending on the user details. When a user views a page with a portlet instance on it, security managers determine whether the user has the appropriate privileges to see the portlet. Implementing access control methods in the producer restricts the retrieval of content from a portlet (that is, hides the portlet) from users without the appropriate privileges. Only if the specified characteristics, such as user details and preferences, pass the authorization logic will the content be retrieved for the user. If no portlet security methods are implemented in the producer, then any user name may be passed in, even fictitious, unauthenticated ones.

`AuthLevelSecurityManager` has access to the following information about authorization level:

- Strongly authenticated.

  The user has signed into the Framework application, and requested the portlet in the context of that session.

- Public or not authenticated.

  The user has not logged in within the context of the current session, and does not have a persistent cookie to indicate that such a state previously existed.

To incorporate these security services into your Java portlet, you must update `provider.xml` and set the security level to strong, weak, or public. Place the following XML right before the `</portlet>` tag in `provider.xml`:

```
<securityManager class="oracle.portal.provider.v2.security.AuthLevelSecurityManager">
   <securityLevel>strong</securityLevel>
</securityManager>
```

After you make this change to `provider.xml`, refresh the producer.



For more information about the syntax of `provider.xml`, see the producer JavaDoc on OTN:

http://www.oracle.com/technology/products/ias/portal/html/javadoc/xml_tag_reference_v2.html

69.18.4.1 Implementing Your Own Security Manager

If your portlet requires special security arrangements which are not provided by the implementations shipped with the PDK, then you must supply your own custom `PortletSecurityManager` controller class. To do this, extend the `oracle.portal.provider.v2.security.PortletSecurityManager` class and supply implementations for the two methods specified by the interface. Then replace the class attribute of the `securityManager` controller element in the XML producer definition with you new class name and configure child elements appropriately.

69.18.5 Message Authentication

Message authentication uses a digital signature. The signature is generated using a Hashed Message Authentication Code (HMAC) algorithm and is based on user information, the shared key, and a UTC (Universal Time, Coordinated) timestamp. The producer authenticates the message using its own copy of the shared key. This technique can be used in Secure Sockets Layer (SSL) communication with a producer instead of client certificates.

For caching purposes, show request signatures are calculated each time a session is set up so producers using message authentication must always have Producer Sessions enabled. This also means that there is a trade-off between performance and security. Shorter session timeouts mean less chance of a message being re-sent illegally, but there is a performance overhead associated with reestablishing a provider session.

A single producer instance cannot support multiple shared keys because it could cause security and administration problems. For instance, if one copy of the shared key is compromised in some way, then the producer administrator has to create a key and distribute it to all of the application clients, who then must update their producer definitions. The way around this problem is to deploy different producer services, specifying a unique shared key for each service. Each producer service has its own deployment properties file so that each service is configured independently of the others. The overhead of deploying multiple producer services within the same producer adapter is relatively small.

While the signature element provides protection against interception and resending of messages, it does nothing to prevent interception and reading of message contents. Messages are still transmitted in plain text. If you are concerned about the content of messages being read by unauthorized people, then this must used in conjunction with SSL to encrypt the message.

The advantage of message authentication is as follows:

- Ensures that the message received by a producer comes from a legitimate WebCenter Portal: Framework application instance.

The disadvantages of message authentication are as follows:

- Causes administration problems if a producer serves multiple portals.
- Entails performance implications if made very secure by having a short session timeout.

69.18.6 User Input Escape

By accepting user input without escaping it to text, you run the risk of an XSS attack, where an attacker attempts to pass in malicious scripts through user input forms. For example, if a portlet title is customizable, then an attacker might attempt to pass scripts or commands to the portlet through the title string. PDK-Java provides the following features to ensure that you can protect your portlets from such attacks:

- Default Container Encoding
- Escape Methods

69.18.6.1 Default Container Encoding

To prevent any script inside a portlet title from being executed, the framework default container renderer class encodes any script characters. This default behavior is controlled through a JNDI variable, `escapeStrings`. When set to `true`, the markup tags in portlet titles are rendered as visible tag characters. For example, a title customization of `<i>title</i>` will be rendered as `<i>title</i>` not *title*. This mode is secure, but, if it is not the desired behavior, then you can set `escapeStrings` to `false` for that producer.

`escapeStrings` applies to all logical producers within a producer. You can set the value of `escapeStrings` from the Fusion Middleware Control Console as you would any other JNDI variable. See Section 61.3.5.2, "Setting JNDI Variable Values" for more information.

69.18.6.2 Escape Methods

If you have code that renders customized values, then you must ensure that you escape those input values appropriately to avoid XSS attacks. This requirement applies to code for rendering pages in any mode. PDK-Java supplies two new static methods for this purpose. They are in the Java class `oracle.portal.provider.v2.url.UrlUtils`, and can be described as follows:

- `public static escapeString(`*`string_text`*`)` escapes any script characters in a given string. For example, less than < becomes `&lt`. This method is unaffected by the `escapeStrings` JNDI variable and is the secure, recommended method to use.
- `public static escapeStringByFlag(`*`string_text`*`)` escapes any script characters in a given string. This method is controlled by the `escapeStrings` JNDI variable and is therefore less secure and not the recommended method to use.

For example:

```
title = UrlUtils.escapeString(data.getPortletTitle());
```

69.19 Troubleshooting Security Issues

This section provides troubleshooting information to help diagnose security related issues for Framework applications.

This section includes the following troubleshooting notes:

- Section 69.19.1, "Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed"

69.19.1 Error Message Appears When Running a Page with a Content Repository Data Control Method Being Consumed

Problem

When you run a page containing a content repository data control method, the following error message appears:

```
"Unable to locate the credential for key extapp in JPS credential store."
```

Cause

Credentials need to be provisioned explicitly using the change password task flow before you access the page that uses the data control method.

Solution

Since the data control is only a model and cannot do anything at the user interface level to allow credential provisioning, you must write an error handler that takes care of the redirection in the user interface. For information about writing an error handler, see the "Customizing Error Handling" section in the Oracle Application Development Framework Developer's Guide.

The following is a sample error handler:

```
public class ErrorHandler
  extends DCErrorHandlerImpl
{
  public ErrorHandler(boolean b)
  {
    super(b);
  }

  public ErrorHandler()
  {
    super(true);
  }

@Override
  public void reportException(DCBindingContainer formBnd, Exception ex)
  {
   if (ex instanceof AdapterException)
    {
      AdapterException ae = (AdapterException) ex;
      if (ae.getCause() != null &&
          ae.getCause() instanceof ExtAppLoginException)
      {
        ExtAppLoginException eale = (ExtAppLoginException) ae.getCause();
        Throwable t = eale.getCause();
         if (t != null &&
            (t instanceof ExtAppCredentialNotFoundException) ||
```

```
            (t instanceof ExtAppInvalidUserCredential))
        {
          String extAppId = eale.getExternalAppId();
          showCredentialsProvisioningDialog(extAppId);
          return;
        }
      }
    }
    super.reportException(formBnd, ex);
  }

  private void showCredentialsProvisioningDialog(String extAppId)
  {
    FacesContext context = FacesContext.getCurrentInstance();
    // Create the dialog UIViewRoot
    ViewHandler viewHandler = context.getApplication().getViewHandler();
    UIViewRoot dialog =
      viewHandler.createView(context,
      "/oracle/adfinternal/extapp/view/pages/CredentialProvisionerDialog.jspx");

     HashMap<String, Object> properties = new HashMap<String, Object>();
     properties.put("width", new Integer(500));
     properties.put("height", new Integer(350));

     HashMap<String, Object> parameters = new HashMap<String, Object>();
     parameters.put("oracle.extapp.id", extAppId);

     RequestContext reqContext = RequestContext.getCurrentInstance();
     //launched from the button, need to specify this for the return listener
     //to be called.
     reqContext.launchDialog(dialog, parameters, null, true, properties);
  }
}
```

70

Deploying and Testing Your WebCenter Portal: Framework Application

This chapter describes how to deploy and test WebCenter Portal: Framework applications using the Integrated WebLogic Server (Integrated WLS) that comes packaged with Oracle JDeveloper and is configured as the default server (DefaultServer). This chapter also describes how to use JDeveloper to deploy applications to an Oracle WebLogic Managed Server configured to test application deployment or host applications for production.

For information about deploying WebCenter Portal: Framework applications through other mechanisms, such as Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Administration Console, and WebLogic Scripting Tool (WLST) commands, see the chapter, "Deploying WebCenter Portal: Framework Applications" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This chapter includes the following sections:

- Section 70.1, "Introduction to Oracle WebLogic Servers"
- Section 70.2, "Deploying a Framework Application to the Integrated WebLogic Server"
- Section 70.3, "Deploying a Framework Application to a WebLogic Managed Server"
- Section 70.4, "Transporting Customizations Between Environments"

70.1 Introduction to Oracle WebLogic Servers

This section provides an overview of the Integrated WebLogic Server (or Integrated WLS) and Oracle WebLogic Managed Servers. Using Oracle JDeveloper, you can test your applications on the Integrated WLS and deploy applications to an Oracle WebLogic Managed Server that resides outside Oracle JDeveloper for staging and further testing or, if you have the required permissions, to an actual production environment.

Integrated WebLogic Server

The Integrated WebLogic Server (IntegratedWebLogicServer) comes packaged with Oracle JDeveloper. The Integrated WebLogic Server connection appears as IntegratedWebLogicServer in the Resource Palette under **IDE Connections > Application Server**. With the Integrated WLS, you can quickly and easily test your application at design time without needing to package the application and create and configure a WebLogic Managed Server.

When you run an application, an Integrated WLS instance for that application automatically starts. You can also start the Integrated WLS manually by choosing the **Run > Start Server Instance** menu option in JDeveloper. Doing so creates a single server instance for all applications; any applications you run will then use this server instance.

Testing WebCenter Portal: Framework applications on the Integrated WLS helps to improve design-time tasks by:

- Optimizing deployment: the requirement to archive and copy files to a separate server for design time testing is eliminated.
- Instantly refreshing changes to the application: you can directly run most files from project directories, which enables you to make selected changes and refresh these changes while the application is running. For example, you can modify a task flow on a JSPX page, then refresh the page in the browser to view your changes without redeploying the application.
- Removing the requirement to undeploy applications after testing and debugging. If you stop the Integrated WLS, however, applications running on it are automatically undeployed.

To learn more about testing a WebCenter Portal: Framework application on the Integrated WLS, refer to Section 70.2, "Deploying a Framework Application to the Integrated WebLogic Server."

WebLogic Managed Server

An Oracle WebLogic Managed Server resides outside of Oracle JDeveloper as part of a domain, and is managed by an Administration server within that domain. A WebLogic Managed Server hosts applications, along with the libraries and other resources needed by those applications. A domain, which is a logically related group of Oracle WebLogic Server resources, can have any number of Managed Servers. Managed Servers can be configured to run applications in a test environment, a production environment, or both.

To learn more about deploying a WebCenter Portal: Framework application to an Oracle WebLogic Managed Server, see Section 70.3, "Deploying a Framework Application to a WebLogic Managed Server."

70.2 Deploying a Framework Application to the Integrated WebLogic Server

The Integrated WLS is preconfigured so that you can run applications within Oracle JDeveloper without needing to create deployment profiles.

However, when the Integrated WLS instance stops, the application is undeployed and therefore becomes unavailable. For a more persistent testing scenario, you can also deploy your application to the Integrated WLS. These two options are described in the following subsections:

- Section 70.2.1, "Running a Framework Application in the Integrated WebLogic Server"
- Section 70.2.2, "Deploying a Framework Application to the Integrated WebLogic Server"

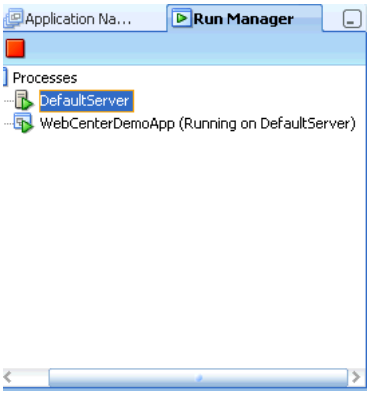
70.2.1 Running a Framework Application in the Integrated WebLogic Server

To run a WebCenter Portal: Framework application, you can right-click the `.jspx` page in the project folder in the Application Navigator and choose **Run**, or select **Run** from the Run menu or click the **Run** icon from the toolbar to run the current project.

When you run an application in JDeveloper, a server instance named after the application is automatically created. While the application is running, you can switch back and forth between JDeveloper and your browser to make changes at design time in your application, save the changes and then refresh your JSPX page in your browser to view those changes. While in JDeveloper, you can also watch the progress of your applications, as well as stop the server instance (and thus stop the application).

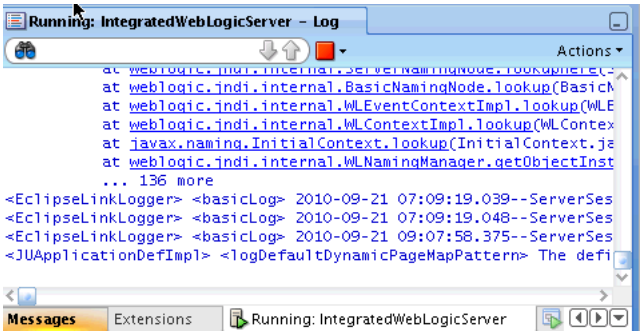
When you run the application page or project, it triggers the packaging and deployment of your WebCenter Portal: Framework application on an Integrated WLS instance named after the application. You can stop instances of the Integrated WLS by using the Run Manager panel (shown in Figure 70–1).To stop an instance, just select **DefaultServer** and click the red Stop icon on the Run Manager tab.

Figure 70–1 Run Manager



Use the DefaultServer Log window (shown in Figure 70–2) to view how the activity is progressing.

Figure 70–2 DefaultServer Log Window



Note: Secure attributes and credentials are migrated by default when running a WebCenter Portal: Framework application on the Integrated WLS.

70.2.2 Deploying a Framework Application to the Integrated WebLogic Server

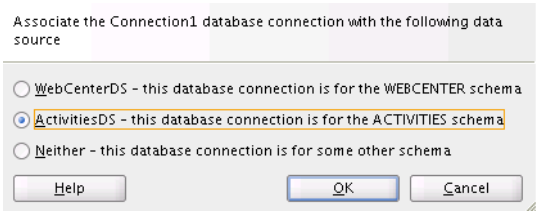
When the Integrated WLS instance stops, the application is undeployed, and therefore, becomes unavailable. For a more persistent testing scenario, you can deploy your application to the Integrated WLS by right-clicking the application and then selecting **Deploy**. This deploys the application to the Default Server instance and it will always be available while the Default Server is running. If you choose this method, then you must first create deployment profiles, as described in Section 70.3.2.2, "Creating Deployment Profiles." If you deploy your application to the Integrated WLS, then the Deployment Configuration dialog displays to let you configure and customize deployment settings. The file system MDS repository pre-created by JDeveloper displays in the **Repository Name** field.

To deploy and run an application in the Integrated WLS:

1. Open your application in JDeveloper.
2. Create the local data source for WebCenter Portal.

   For more information about how to create a local data source, see Section 7.2.2, "Setting Up a Database Connection."
3. If you're using Analytics task flows, create a local data source for Activities.
   a. From the File menu, select **New**.
   b. From the Connections category, select **Database Connection** and click **OK**.
   c. Enter the connection details for the connection to the Activities schema and click **OK**.
   d. Associate the connection with the ActivitiesDS data source as shown in Figure 70–3 and click **OK** .

Figure 70–3 Associate to Data Source Dialog - Activities Schema



4. From the Run menu, select **Run xxxx.jpr** (where **xxxx** is the name of your project) to deploy and run your application.

70.3 Deploying a Framework Application to a WebLogic Managed Server

When you're ready to test your Framework application using an external test or staging site, you can deploy your applications directly from JDeveloper to an Oracle WebLogic Managed Server.

For information about deploying Framework applications using Fusion Middleware Control, Oracle WebLogic Administration Console, and WLST commands, see the chapter, "Deploying WebCenter Portal: Framework Applications" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section includes the following:

- Section 70.3.1, "Deployment Roadmap"

- Section 70.3.2, "Packaging a Framework Application"
- Section 70.3.3, "Preparing the Target Environment for Deployment"
- Section 70.3.4, "Deploying a Framework application to a Managed Server"

70.3.1 Deployment Roadmap

The flow chart and table in this section provide an overview of the prerequisites and tasks required to deploy a Framework application to an Oracle WebLogic Managed Server. Figure 70–4 shows the steps to deploy a Framework application, and the roles that will carry them out.

Figure 70–4 Deploying a Framework Application to a Managed Server

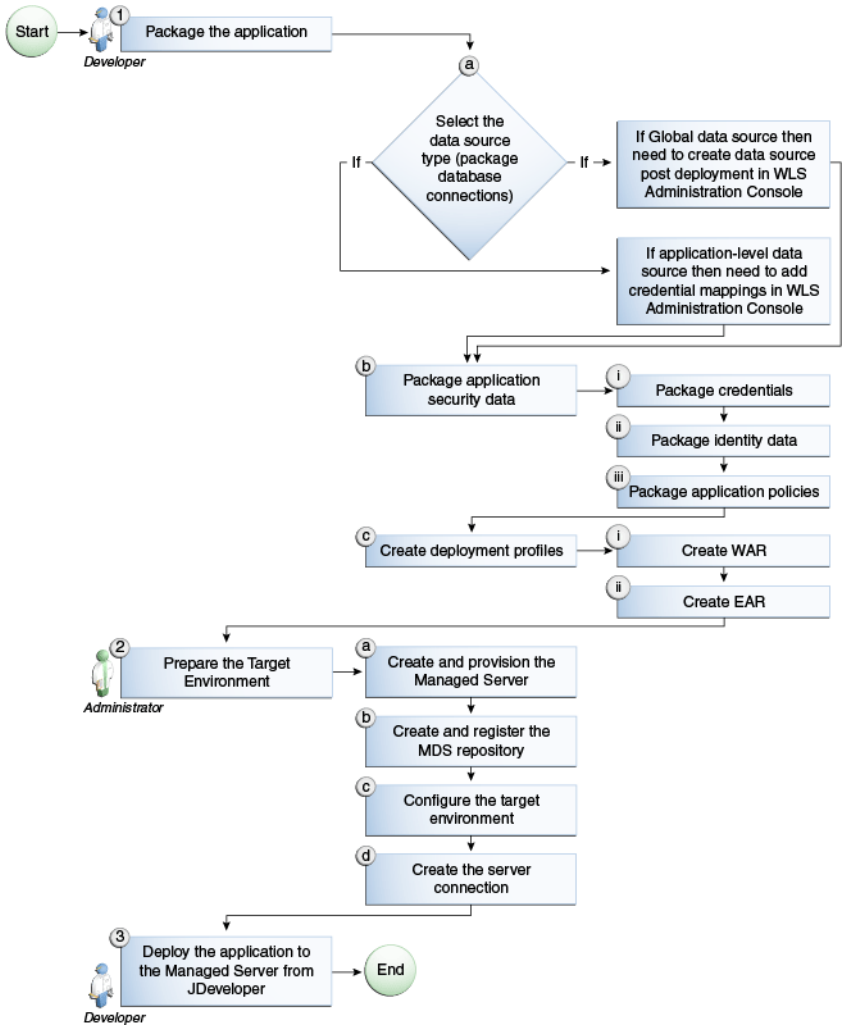


Table 70–1 shows the tasks, sub-tasks and who will need to carry them out to deploy a Framework application from JDeveloper.

Table 70–1 Deploying a Framework Application to a Managed Server

Actor	Task	Sub-task	Notes
Developer	**1.** Package the Application	**1.a** Select the data source type (package database connections)	You can use either a global data source or an application-level data source. If using a global data source, then you need to create the data source in the WLS Administration Console before deploying. If using an application-level data source, then you may need to add credential mappings using the WLS Administration Console after deploying (see the note below). **Note:** There are two options for deploying from JDeveloper. You can deploy directly to the Managed Server, or you can deploy to an EAR file and then deploy the EAR file using either Fusion Middleware Control, WLST or the WLS Admin Console. If you deploy directly from JDeveloper the credential mapping step is not required as JDeveloper automatically handles the credential mapping. However, if you deploy the EAR file outside of JDeveloper, then this step is required.
		1.b Package application security data	This sub-task consists of packaging the credentials, identity data, and application policies.
		1.c Create deployment profiles	This sub-task consists of creating the WAR and EAR files.
Administrator	**2.** Prepare the Target Environment	**2.a** Create and provision the Managed Server	
		2.b Create and register the MDS repository	
		2.c Configure the target environment	
		2.d Create the server connection	
Developer	**3.** Deploy the Application to a Managed Server from JDeveloper		The final step is to deploy the application to the Managed Server from JDeveloper.

70.3.2 Packaging a Framework Application

When you deploy WebCenter Portal: Framework applications, you may also need to migrate your database connections and the related security for connecting to the databases. You will also need to create a set of project-level and application-level deployment profiles that indicate how a WebCenter Portal: Framework application and its associated files should be packaged so that the application can be deployed to an Oracle WebLogic Managed Server.

This section includes the following:

- Section 70.3.2.1, "Packaging Database Connections and Application Security"
- Section 70.3.2.2, "Creating Deployment Profiles"

70.3.2.1 Packaging Database Connections and Application Security

When you deploy WebCenter Portal: Framework applications, you'll also need to package your database connections and the related security for connecting to the databases. This section describes how to package a WebCenter Portal: Framework application that uses one of the two database connection types: JDBC data source and JDBC URL. This section also describes security migration for applications that use database connections that specify passwords, or external applications that specify shared or public credentials.

Based on the policy defined by your administrator (if, for example, your administrator does not allow any application-level data sources), you may need to migrate the database connections and respective credentials to the target server. The packaging options described below allow you to choose how the database connections should be treated.

This section includes the following sub-sections:

- Section 70.3.2.1.1, "Packaging the Database Connections"
- Section 70.3.2.1.2, "Packaging Application Security Data"

70.3.2.1.1 Packaging the Database Connections

If your WebCenter Portal: Framework application contains JDBC database connections, you'll need to choose how JDeveloper will migrate those database connections to the Oracle WebLogic Managed Server. You can choose from:

- *Global data source*:

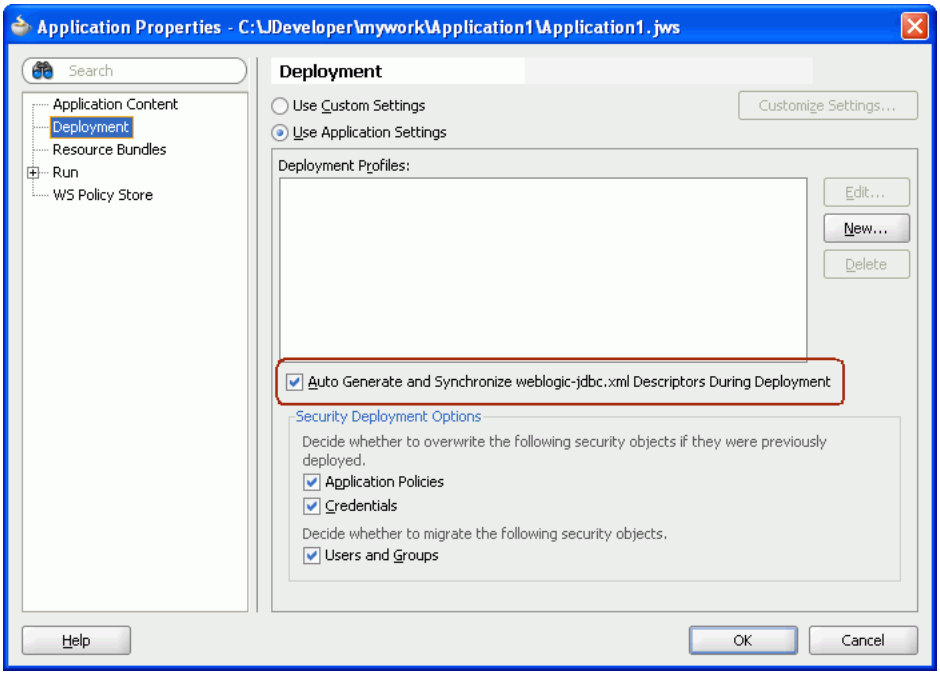
  Oracle recommends that you choose this type of data source if you plan to deploy an application EAR file to a Managed Server running in production mode using Fusion Middleware Control, the Oracle WebLogic Administration Console, or using a WLST (`wldeployer`) command.

  To choose a global data source, deselect the **Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment** checkbox when you create the EAR file so that JDeveloper does not create an application-level data source with password indirection. Instead, create the global data source using Oracle WebLogic Administration Console prior to or after deploying the application to a Managed Server in production mode using the default WebCenter Portal and Activities schemas. For more information on creating global data sources, see the section "Creating a JDBC Data Source" in *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

- *Application-level data source with password indirection*:

  JDeveloper generates an application-level data source with password indirection by default when you run an application on the Integrated WLS. That is, when the **Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment** checkbox is selected in the Application Properties dialog, as shown in Figure 70–5.

Figure 70–5 Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment Checkbox



To generate an application-level indirection data source, JDeveloper does the following:

- Generates a `<connection>-jdbc.xml` file (such as `WC-jdbc.xml`) for each connection in the Application Resource.
- Sets the indirect password attribute in the `<connection>-jdbc.xml` file:

```
<jdbc-driver-params>
<use-password-indirection>true</use-password-indirection>
</jdbc-driver-params>
```

- Updates `weblogic-application.xml` to add each `<connection>-jdbc.xml` file as a module. For example:

```
<module>
<name>WC</name>
<type>JDBC</type>
<path>META-INF/WC-jdbc.xml</path>
</module>
```

- Adds a resource reference to each JDBC JNDI name in the `web.xml` file, if this file exists. For example:

```
<resource-ref>
<description>WC-Connection</description>
<res-ref-name>jdbc/WCDS</res-ref-name>
<res-type>javax.sql.DataSource</res-type>
<res-auth>Container</res-auth>
</resource-ref>
```

If you choose to generate an application data source with password indirection, then you must add credential mappings using the Oracle WebLogic Administration Console to be able to activate the application. For more information on adding credential mappings, see the section "Creating a JDBC Data

Source" in *Oracle Fusion Middleware Configuring and Managing JDBC Data Sources for Oracle WebLogic Server*.

Note: Do not use WLST commands to deploy applications that use application-level data sources with password indirection, as WLST cannot set password indirection credential mappings on the server.

70.3.2.1.2 Packaging Application Security Data

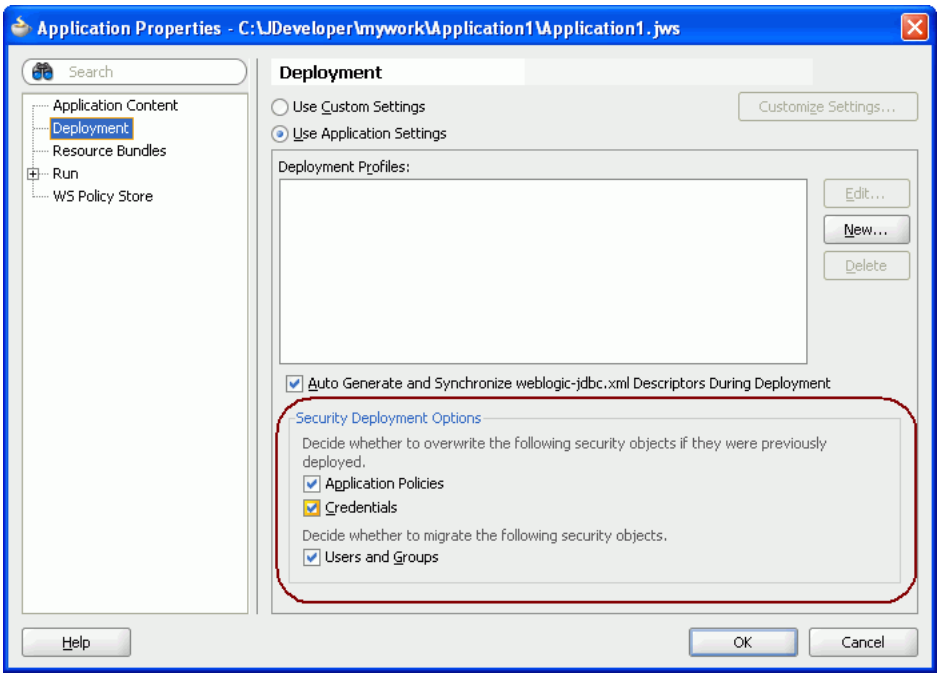
The following is applicable only if you have configured security in your application using the Configure ADF Security wizard, or the application contains connections with secure attributes such as a database connection password or external applications with shared or public credentials:

- Packaging Credentials
- Packaging Identity Data
- Packaging Application Policies

Packaging Credentials

If you do not intend to migrate credentials, then deselect the **Credentials** checkbox in the **Security Deployment Options** section in the Application Properties dialog (see Figure 70–6).

Figure 70–6 Security Deployment Options



If you retain the default selection in the Security Deployment Options section, then the credential migration will behave as follows, depending on whether the WebLogic domain is in development or production mode:

- When you deploy a WebCenter Portal: Framework application to a Managed Server running in production mode, secure attributes of the connections packaged within your application and any shared or public credentials specified for external

applications are not migrated to the domain-level credential store. This is because secure properties are likely to be different between development and production environments. Therefore, you must reconfigure secure attributes of connections and shared or public credentials for external applications using WLST commands or Fusion Middleware Control, as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

For more information on credential migration behavior, see the section "Configuring the Credential Store" in *Oracle Containers for J2EE Security Guide*.

- When you deploy a Framework application in development mode, you must first enable credential migration. To enable credential migration for an application to be deployed in development mode, start the Managed Server with `-Djps.app.credential.overwrite.allowed=true`, by adding the following option to the `setDomainEnv.cmd` or `setDomainEnv.sh` file located in *`domain`*`\bin`:

```
set EXTRA_JAVA_PROPERTIES=-Djps.app.credential.overwrite.allowed=true %EXTRA_
JAVA_PROPERTIES%
```

Packaging Identity Data

In Figure 70–6, in the Security Deployment Options section, the Users and Groups checkbox is selected by default. This indicates that the users and groups defined in the application's `jazn-data.xml` file will be migrated to the identity store configured for the WebLogic Managed Server, if the authenticator configured in the domain allows creation of users and groups. When migrating the identify store, you should deselect this option as users and groups defined in the identity store configured for the Managed Server should be used to access the application.

Packaging Application Policies

When the **Application Policies** checkbox is selected, then the `jps.policystore.migration` parameter in the `weblogic-application.xml` file is set to `OVERWRITE`. This means that during application deployment and redeployment, the application policies packaged with the application will be overwritten on the domain-level policy store for the domain to which the application is being deployed.

If the **Application Policies** checkbox is not selected, then the `jps.policystore.migration` parameter is set to `MERGE`. This means that the application policies will be migrated during application deployment but not during redeployment, and the existing application policies will not be overwritten.

70.3.2.2 Creating Deployment Profiles

Deploying your application to a Managed Server that resides outside JDeveloper using Fusion Middleware Control or the WLS Administration Console can only be done using a deployment profile, or Enterprise Archive (EAR) file. The EAR file packages (or archives) a WebCenter Portal: Framework application and its associated files so that the application can be deployed to an Oracle WebLogic Managed Server (this is typically done by a system administrator when the target is a production server).

> **Note:** You can deploy Oracle ADF applications, such as WebCenter Portal: Framework applications, only as EAR files.

The EAR file contains application artifacts such as, `adf-config.xml`, `connections.xml`, `weblogic-application.xml`, `jazn-data.xml`, metadata

archive (MAR) files, and all project-related WAR files. For WebCenter Portal: Framework applications containing MDS metadata or portlets, a metadata archive file (with the extension .MAR) is automatically generated and included in the EAR file. This file contains all MDS metadata and portlet customization data.

For Framework applications, both the WAR (project-level) and EAR-level deployment profiles are created for you as you build your application. These are named:

```
<application name>_webapp1
<application name>_application1
```

respectively, where *<application name>* is the name given to the application at creation time. Note that this only applies to Framework applications created based on the WebCenter Portal template.

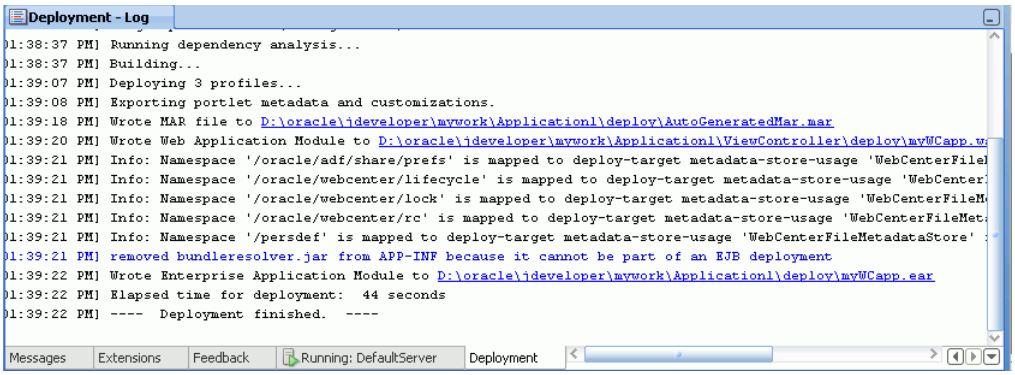
If your application, however, was not created using the WebCenter Portal template, then you will need to create a deployment profile file for your application. You may also want to create your own EAR file if you want to give it a specific name or change what's included in the EAR file through the Application Assembly section.

To create an EAR file:

1. In JDeveloper, in the Application Navigator, right-click the application and select **New**.
2. In the New Gallery, expand **General**, select **Deployment Profile**, then **EAR File**, and click **OK**.
3. In the Create Deployment Profile – EAR File dialog, enter a name for your deployment profile and click **OK**.
4. In the Edit EAR Deployment Profile Properties dialog, select **Application Assembly**, then select the project WAR you want to make available in the project file and and click **OK**.
5. In the Application Properties dialog, select the deployment options to use and click **OK**.

   For a description of the **Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment** option, see Section 70.3.2.1.1, "Packaging the Database Connections." For a description of the Deployment Security Options, see Section 70.3.2.1.2, "Packaging Application Security Data."
6. In the Application Navigator, right-click the application name, choose **Deploy**, *deployment profile name*, **to**, and then choose **to EAR file**. This creates the EAR file in the **deploy** folder located in *JDEV_HOME*\mywork*application_name*\deploy\, as shown in Figure 70–7.

Figure 70–7 EAR Status in Deployment - Log



70.3.3 Preparing the Target Environment for Deployment

This section describes how to create and provision a WebLogic Managed Server instance prior to deploying your Framework application.

This section includes the following sub-sections:

- Section 70.3.3.1, "Creating and Provisioning an Oracle WebLogic Managed Server Instance"
- Section 70.3.3.2, "Creating and Registering the Metadata Service Repository"
- Section 70.3.3.3, "Configuring the Target Environment"
- Section 70.3.3.4, "Creating a WebLogic Managed Server Connection"

70.3.3.1 Creating and Provisioning an Oracle WebLogic Managed Server Instance

Before deploying a WebCenter Portal: Framework application, you must create a WebLogic Managed Server based on the "Oracle WebCenter Custom Portal" template that contains all the required shared libraries and a MDS Repository. For instructions on how to create a new managed server, see "Extending an Existing Domain" in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*. For instructions on how to create a new domain, see "Creating a New Domain" in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

> **Note:** Oracle does not recommend deploying Framework applications to any of the three pre-configured Managed Servers created during the installation, or to a domain Administration Server.

70.3.3.2 Creating and Registering the Metadata Service Repository

After creating and provisioning the WebLogic Managed Server instance, you must create and register a Metadata Service Repository (MDS) schema for the application on the WebLogic Domain's Administration Server instance, as described in the section "Creating and Registering the Metadata Service (MDS) Repository" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

70.3.3.3 Configuring the Target Environment

After creating the Managed Server and creating and registering the MDS repository, continue by configuring the data sources, and the connections to the Identity Store, and the Policy and Credential Store. For information on configuring data sources, see the section "Creating a JDBC Data Source" in *Oracle Fusion Middleware Configuring and*

Managing JDBC Data Sources for Oracle WebLogic Server, or How to Create a JDBC Data Source for Oracle WebLogic Server. Note that when setting up the data source, you must provide a password or the connection may not be created when the application is deployed.

70.3.3.4 Creating a WebLogic Managed Server Connection

Before you can deploy your applications using JDeveloper to an Oracle WebLogic Managed Server instance that reside outside JDeveloper, you must create a connection to the Managed Server instance where you will deploy the application. Before you create a connection to an Oracle WebLogic Managed Server instance, ensure that the target Managed Server instance is up and running, and has the required libraries.

To create a connection to an Oracle WebLogic Managed Server:

1. In JDeveloper, from the File menu, select **New**.
2. In the New Gallery, expand General, select **Connections**, and then **Application Server Connection**.
3. Click **OK**.
4. In the Create Application Server Connection wizard, for Step 1, enter a name for the new connection (for example, `WC_CustomPortal`), and then click **Next**.
5. At Step 2, specify the user name and password for authentication, and then click **Next**.
6. At Step 3, enter the host name of the WebLogic Managed Server (for example, `webcenter.myserver.example.com`) and the port number (for example, 7888).
7. In the WLS Domain field, specify the name of the domain in which the WebLogic Managed Server instance is created (for example, `wc_server`), and then click **Next**.
8. At Step 4, click **Test Connection**.

   If the test is successful, you now have a connection to the target WebLogic Managed Server.

70.3.4 Deploying a Framework application to a Managed Server

You can deploy a Framework application to a Managed Server using either local application or global data sources. Local application data sources can only be accessed by the deployed application. However, it's an easy way to deploy in that it needs no other data source configuration in order to work if deployed directly from JDeveloper.

Global data sources benefit from being able to be shared by other applications on the server where database connection config updates can be more easily maintained and picked up by all users of the global data source. For more information about data sources, see the section "Choosing the Data Source" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

This section contains the following subsections:

- Section 70.3.4.1, "Deploying to a Managed Server Using Local Data Sources"
- Section 70.3.4.2, "Deploying to a Managed Server Using Global Data Sources"

70.3.4.1 Deploying to a Managed Server Using Local Data Sources

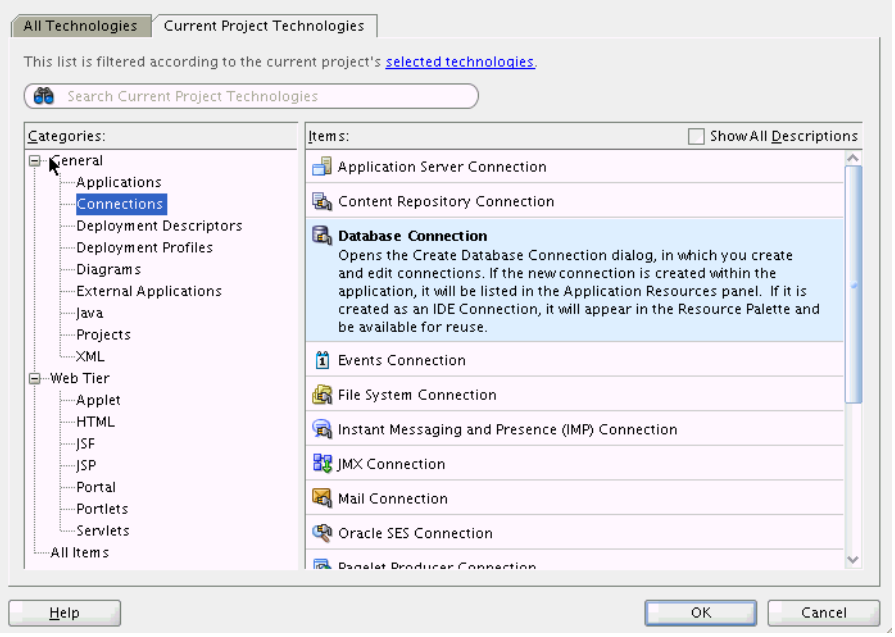
When you deploy to a Managed Server, the application EAR file is generated. The EAR file packages the metadata archive (MAR file), which includes the metadata content to be deployed to an MDS repository. Additionally, the `adf-config.xml` file is reconfigured with a modified `mds-config` for the target deployment environment. Application-wide features, security, caching, and change persistence remain unchanged. Properties for other Oracle components, if any, are also configured in this file. Similarly, JSF and JSTL shared libraries are added in the `weblogic.xml` file during packaging.

To deploy a WebCenter Portal: Framework application to a Managed Server using local data sources:

1. In JDeveloper, in the Application Navigator, open the application to be deployed.
2. Create the local data source for WebCenter Portal (if required).
   a. From the File menu, select **New**.

      The New Gallery appears (see Figure 70–8).

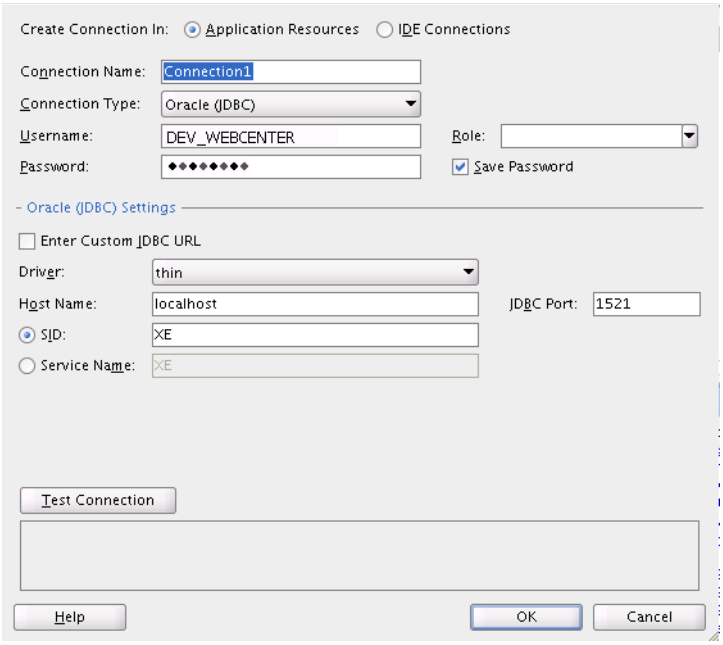
Figure 70–8 New Gallery



   b. From the Connections category, select **Database Connection** and click **OK**.

      The Create Database Connection Dialog displays (see Figure 70–9).

Figure 70–9 Create Database Connection



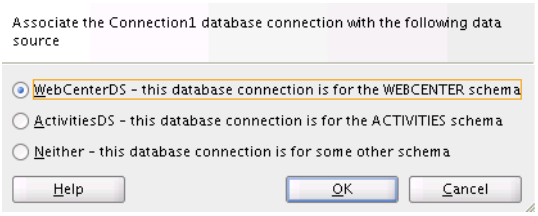
c. Enter the connection details for the connection to the WebCenter Portal schema and click **OK**.

Note: When deploying directly from JDeveloper to a WLS Managed Server the local data source will only be created if the password is saved in the connection (that is, the **Save Password** option is checked), and the database is accessible at deployment time. If there is no saved password, or if the credentials are not able to access a database, the data source will not be created, the application will fail to deploy, and an error will be generated.

When deploying indirectly using Fusion Middleware Control or WLS Administration Console, however, the data sources are created and can be configured subsequent to deployment.

The Associate to Data Source dialog displays (see Figure 70–10).

Figure 70–10 Associate to Data Source Dialog



d. If there are multiple projects select the project, associate the connection with the WebCenterDS data source and click **OK**.

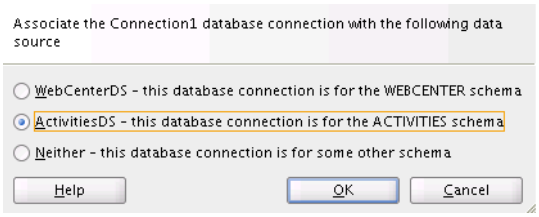
Note that the Associate to Data Source dialog can also be invoked from the context menu for a database connection (right-click on a database connection and select **Associate to Data Source**). This is useful if you change your mind

and want to associate the database connection to a different data source, or disassociate it (by selecting **Neither**).

3. If you're using Analytics task flows, create a local data source for Activities.
   a. From the File menu, select **New**.
   b. From the Connections category, select **Database Connection** and click **OK**.
   c. Enter the connection details for the connection to the Activities schema and click **OK**.
   d. Associate the connection with the ActivitiesDS data source as shown in Figure 70–11 and click **OK** .

      Note that the Associate to Data Source dialog can also be invoked from the context menu for a database connection (right-click on a database connection and select **Associate to Data Source**). This is useful if you change your mind and want to associate the database connection to a different data source, or disassociate it (by selecting **Neither**).

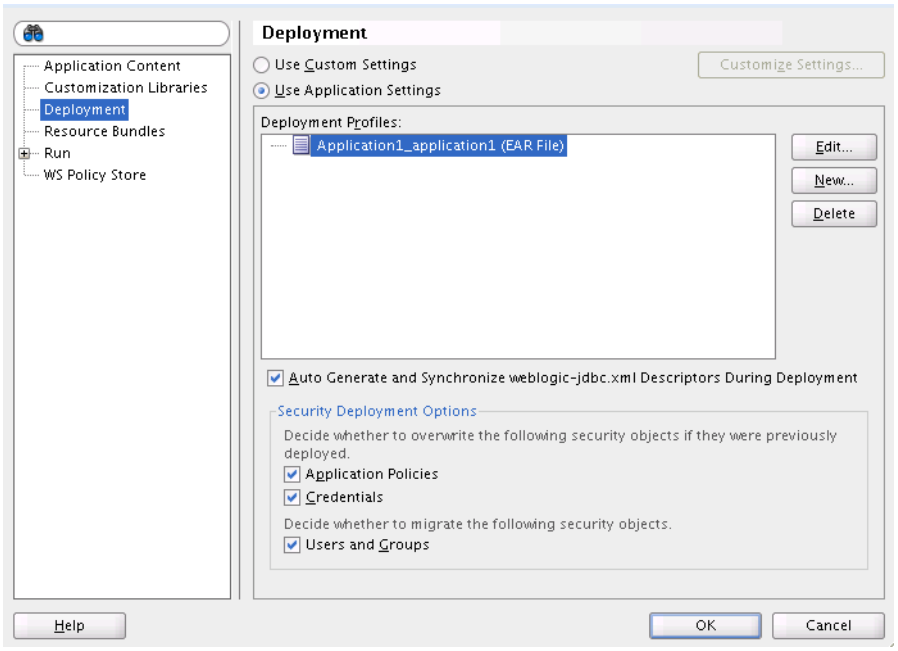
Figure 70–11 Associate to Data Source Dialog - Activities Schema



4. Right-click the Application menu and select **Application Properties...**

   The Application Properties dialog displays (see Figure 70–12).

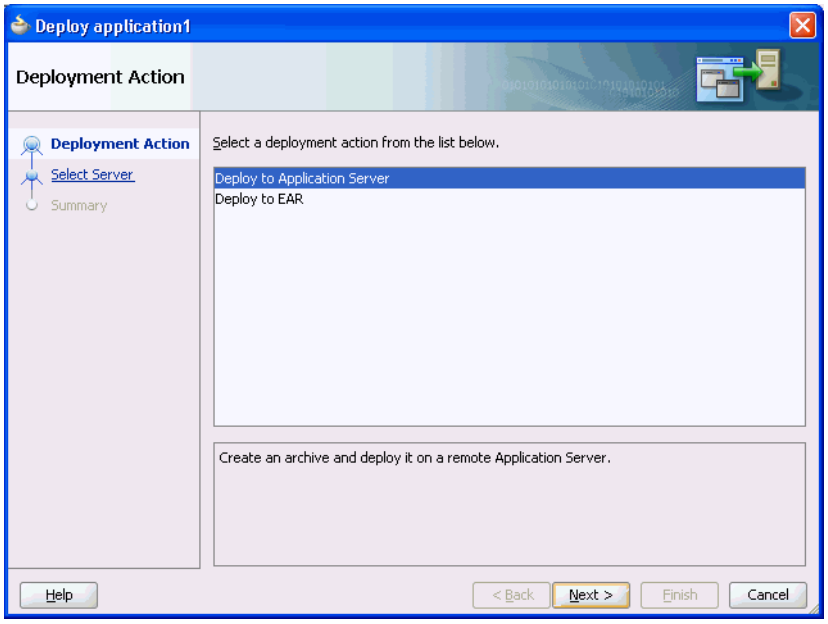
Figure 70–12 Application Properties Dialog



5. Choose **Deployment** from the navigation panel to display the Deployment options.
6. Make sure that the `Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment` option is selected and click **OK**.
7. Right-click on the Application menu, select **Deploy** and then the application name.

   The Deployment Action dialog displays (see Figure 70–13).

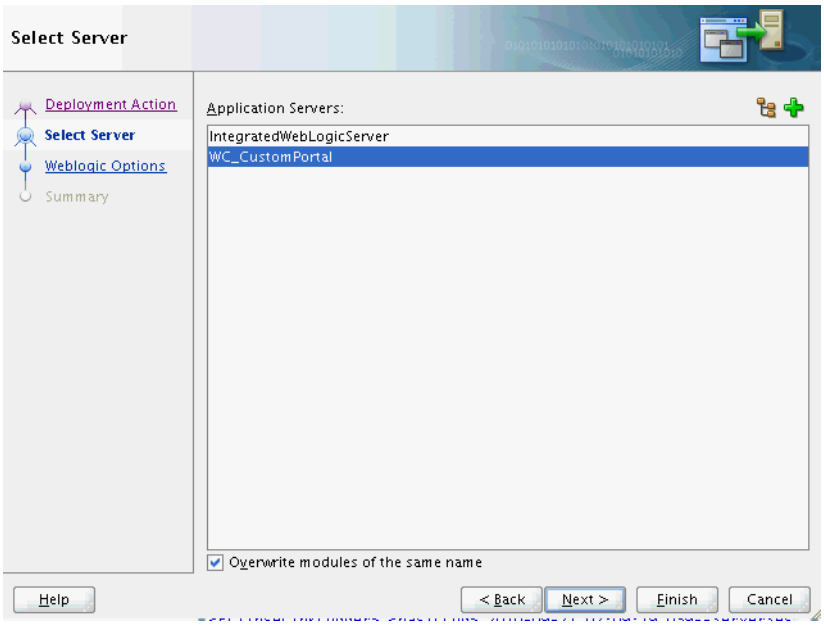
Figure 70–13 Deployment Action Dialog



8. Select **Deploy to Application Server** and click **Next**.

   The Select Server dialog displays (see Figure 70–14).

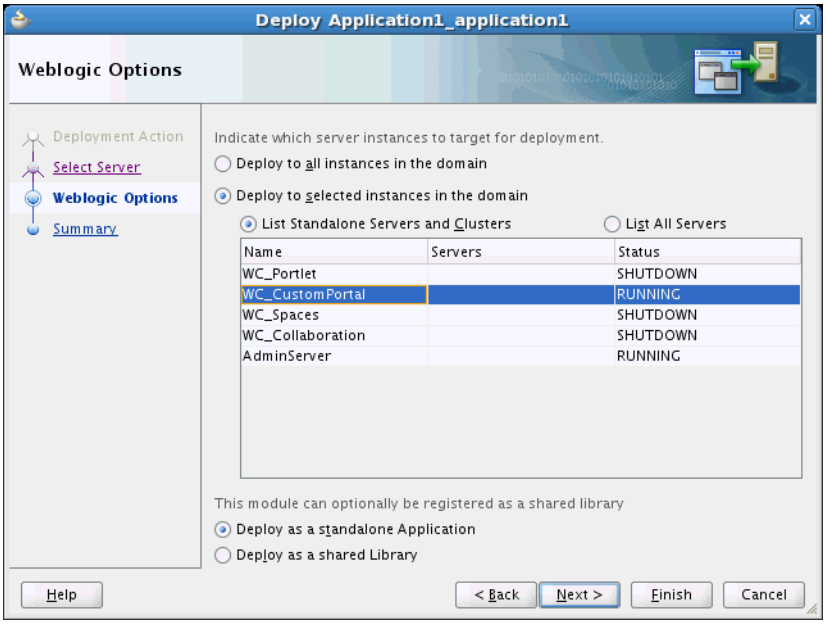
Figure 70–14 Select Server Dialog



9. Select the connection name of the Managed Server (for example, `WC_CustomPortal`), and click **Next**. If the server to which you want to deploy is not listed, click the Add icon (+) and complete the connection details to add the application server to the list.

   The Deployment WebLogic Options dialog displays (see Figure 70–15).

Figure 70–15 Deployment WebLogic Options dialog



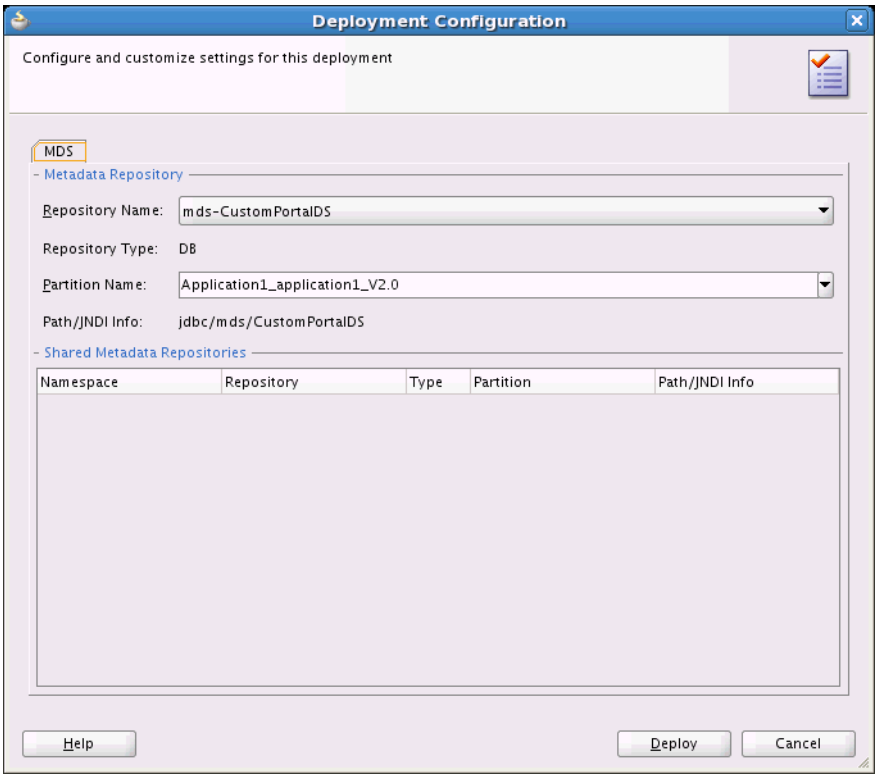
10. Select `Deploy to selected instance in the domain`, select the managed server from the list (for example, `WC_CustomPortal`), and click **Next**.

    In most cases you will want to deselect **Deploy to all server instances in the domain** and select the specific servers and clusters to deploy to using the Server

Instances dialog. Refer to the online help for more information about using the Select Server dialog options.

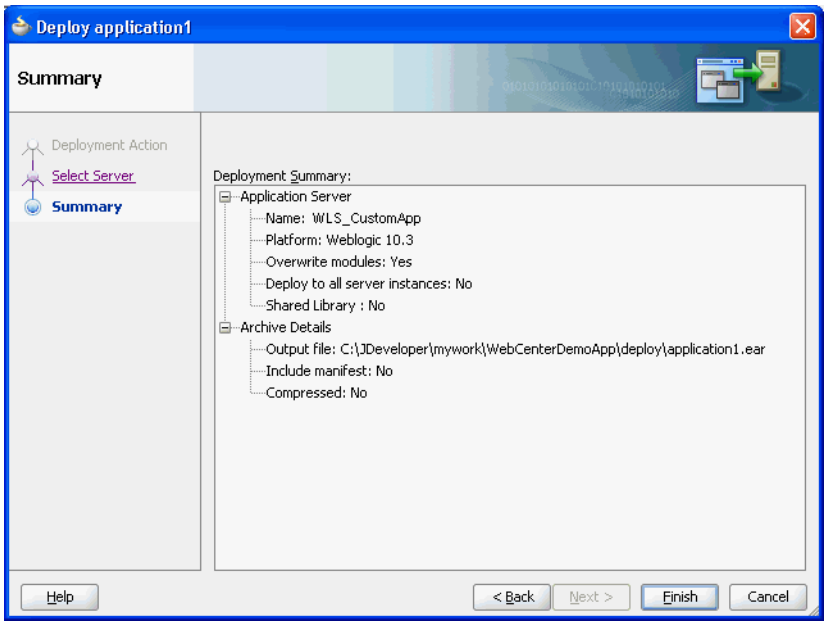
The Deployment Configuration dialog displays (see Figure 70–16).

Figure 70–16 Deployment Configuration Dialog



11. Select the Repository Name (for example, `mds-CustomPortalDS`), enter a Partition Name (if the partition doesn't already exist then it will be created during the deployment), and click **Deploy**.
12. On the Deployment Summary dialog (see Figure 70–17), check that all of the deployment options are correct and click **Finish** to start the deployment.

Figure 70–17 Deployment Summary Dialog



13. Continue by doing any post-deployment security or data source connection configurations that may be required as described in the section on "Post-Deployment Configuration" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

70.3.4.2 Deploying to a Managed Server Using Global Data Sources

Deploying to a Managed Server using global data sources is recommended when the application is not intended to run on a Managed Server created with the WebCenter Custom Portal template, or is intended to run against custom data sources not named `WebCenterDS` or `ActivitiesDS`.

Oracle recommends using the default data source names, but if you use non-default pre-seeded data sources, then you must name your data sources accordingly and follow the pre- or post-deployment global data source creation steps to create data sources that map to the names you have chosen. For more information, see the section "Choosing the Data Source" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To deploy to a Managed Server using global data sources:

1. Open the application to be deployed in JDeveloper.
2. Determine the JNDI names your application will be using for the data-sources.

   The JNDI names will depend on whether you've created a database connection for your application in JDeveloper or are using the defaults. The following table shows the default JNDI names:

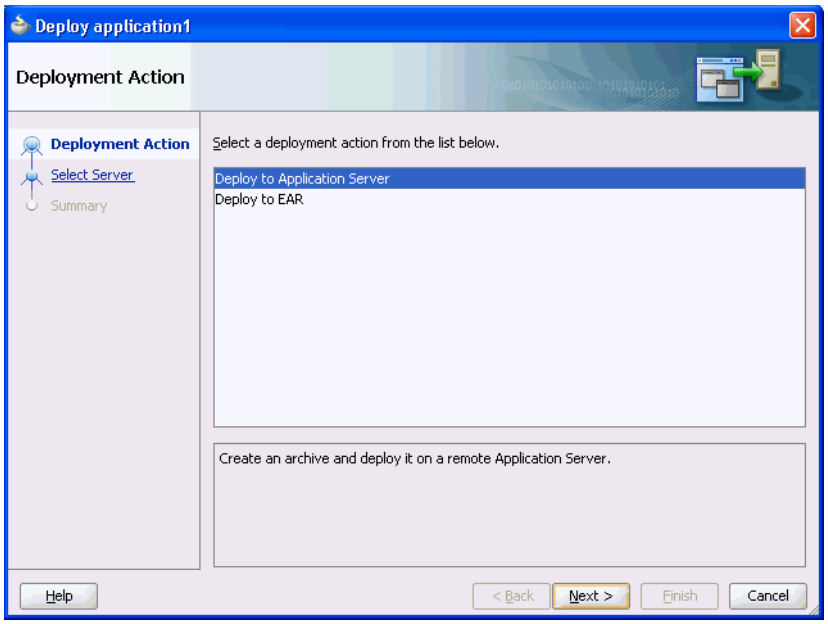
Data Source	JDeveloper JNDI Name	Default JNDI Name
WebCenterDS	`jdbc/`*`WebCenterConnectionNameDS`* (where *`WebCenterConnectionName`* is the name of the WebCenter Portal database connection)	`jdbc/webcenter/CustomPortalDS`

Data Source	JDeveloper JNDI Name	Default JNDI Name
ActivitiesDS	`jdbc/ActivitiesConnectionNameDS` (where `ActivitiesConnectionName` is the name of the Activities database connection)	`jdbc/activities/CustomPortalDS`

3. Choose **Deployment** from the navigation panel to display the Deployment options.
4. Deselect the `Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment` option if selected and click **OK**.
5. Right-click on the Application menu, select **Deploy** and then the application name.

   The Deployment Action dialog displays (see Figure 70–13).

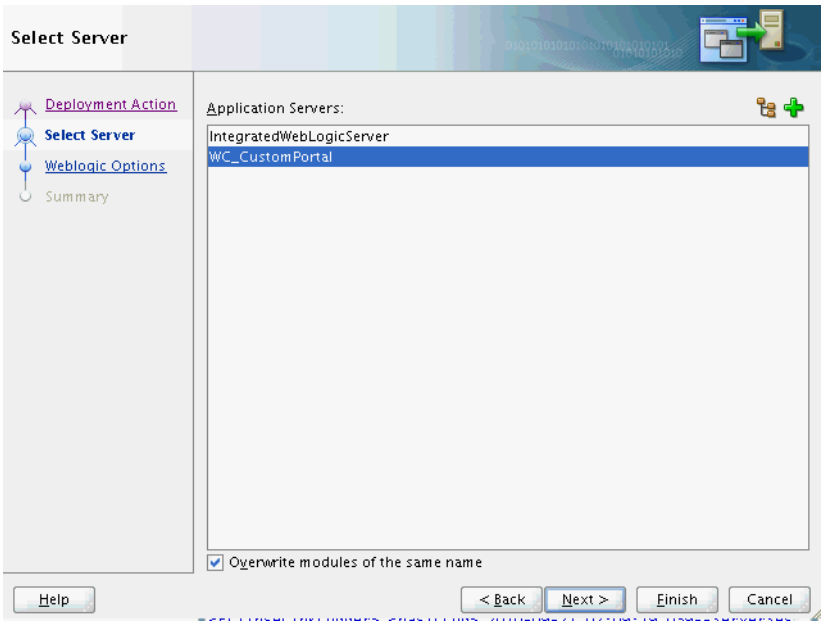
Figure 70–18 Deployment Action Dialog



6. Select **Deploy to Application Server** and click **Next**.

   The Select Server dialog displays (see Figure 70–14).

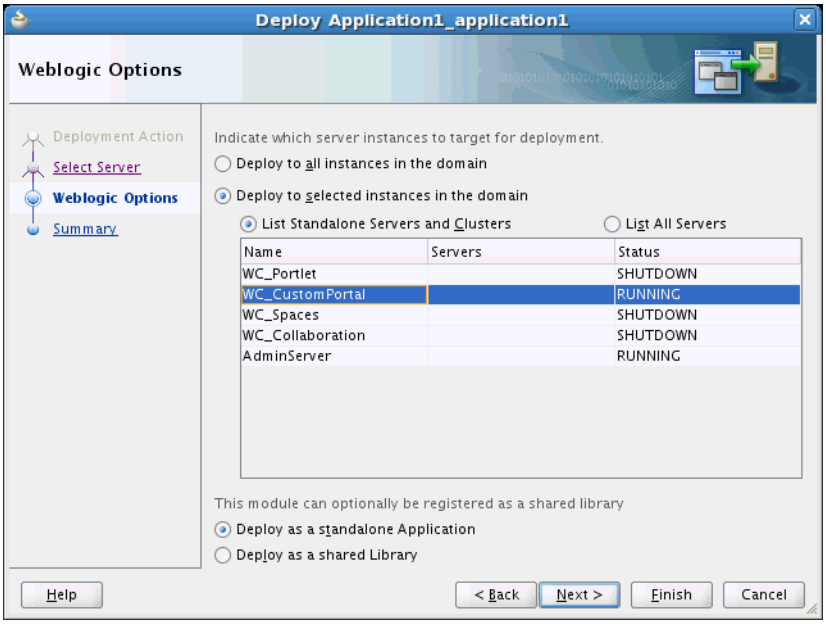
Figure 70–19 Select Server Dialog



7. Select the connection name of the Managed Server (for example, `WC_CustomPortal`), and click **Next**. If the server to which you want to deploy is not listed, click the Add icon (+) and complete the connection details to add the application server to the list.

   The Deployment WebLogic Options dialog displays (see Figure 70–15).

Figure 70–20 Deployment WebLogic Options dialog



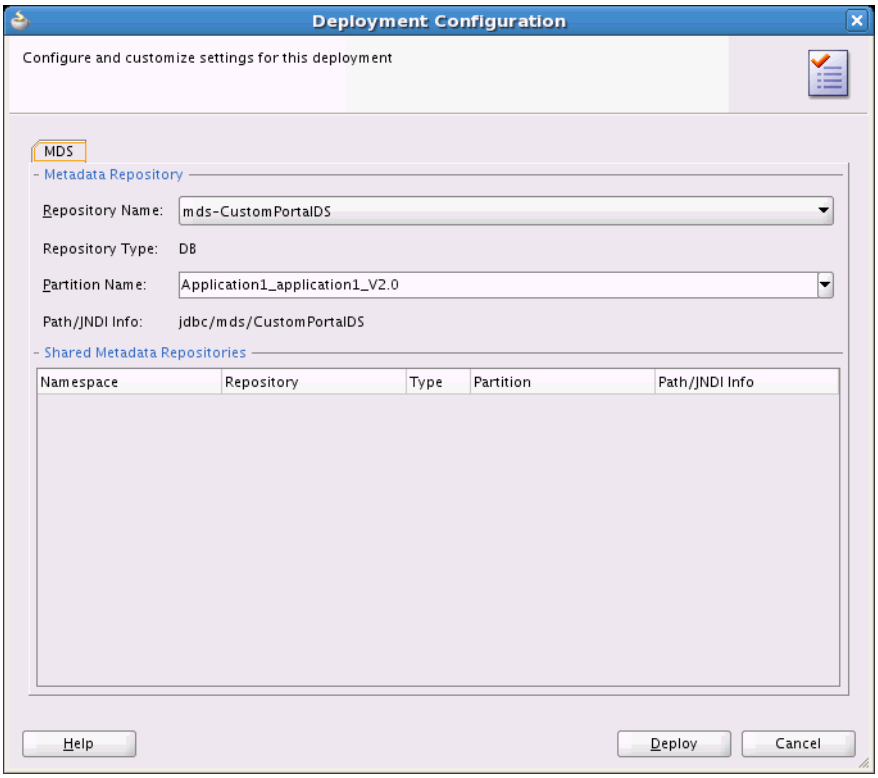
8. Select `Deploy to selected instance in the domain`, select the managed server from the list (for example, `WC_CustomPortal`), and click **Next**.

   In most cases you will want to deselect **Deploy to all server instances in the domain** and select the specific servers and clusters to deploy to using the Server

Instances dialog. Refer to the online help for more information about using the Select Server dialog options.

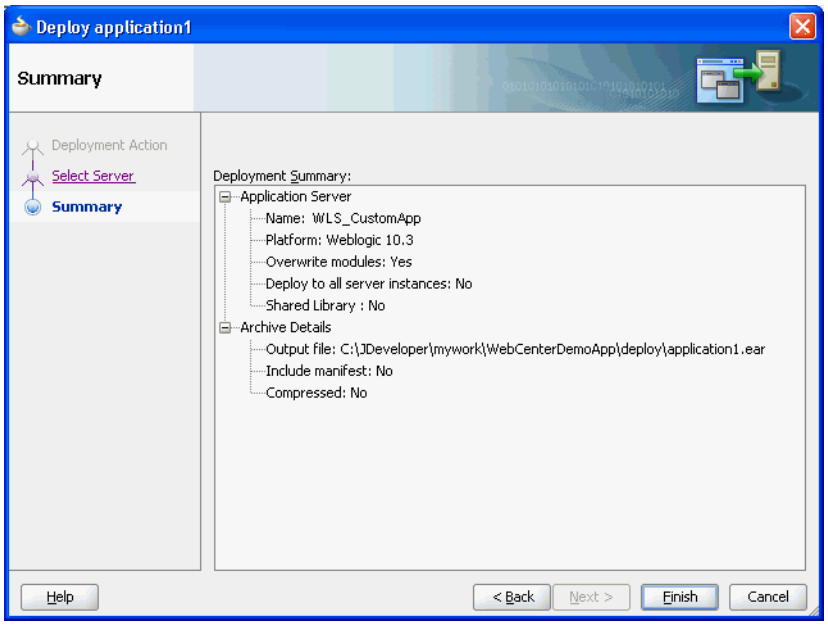
The Deployment Configuration dialog displays (see Figure 70–16).

Figure 70–21 Deployment Configuration Dialog



9. Select the Repository Name (for example, `mds-CustomPortalDS`), enter a Partition Name (if the partition doesn't already exist then it will be created during the deployment), and click **Deploy**.
10. On the Deployment Summary dialog (see Figure 70–17), check that all of the deployment options are correct and click **Finish** to start the deployment.

Figure 70–22 Deployment Summary Dialog



11. Continue by doing any post-deployment security or data source connection configurations that may be required as described in the section on "Post-Deployment Configuration" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

70.4 Transporting Customizations Between Environments

When migrating a deployed application to a new environment, post-deployment customizations made to pages, WebCenter Portal service, and portlets (PDK-Java and WSRP version 2 producers) must be migrated too. Export and import utilities are available to assist with this process. For more information, see the chapter, "Managing Backup, Recovery, Export, and Import of WebCenter Portal" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

71

Building Multilanguage Portals

When your application will be viewed by users in multiple countries, you can configure your JSF page or application to use different locales so that it displays the correct language for the language setting of a user's browser. For example, if you know your page will be viewed in Italy, you can localize your page so that when a user's browser is set to use the Italian language, text strings in the browser page appear in Italian.

Additionally, locale selection applies special formatting considerations applicable to the selected locale. For example, whether information is typically viewed from left to right or right to left, how numbers are depicted (such as monetary information), and the like.

When you develop a multi-language portal:

- Make use of the inherent multi-language support in Oracle WebCenter Portal: Services. For more information, see Section 71.1, "Language Support in ADF Faces Components."
- Put all resources in resource bundles, provide alternate-language resource bundles, and code your application to respond to users' language selection. For more information, see Section 71.2, "Using Resource Bundles to Support Multiple Languages."
- Make sure your database character set supports all required languages.

  For more information, see "Choosing a Character Set" in the *Oracle Database Globalization Support Guide*.
- Use number, date, and time formatting that supports all required locales.

  For more information, see "Setting Up a Globalization Support Environment" in the *Oracle Database Globalization Support Guide*.
- Use linguistic sort parameters to get the correct sort ordering for locale.

  For example, the same characters have different sort ordering in French and Canadian-French.

  For more information, see "Linguistic Sorting and String Searching" in the *Oracle Database Globalization Support Guide*.
- Write web pages that correctly render left to right or right to left (for example, if Arabic languages must be supported).

For more guidelines on globalization, see "Internationalizing and Localizing Pages" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework* or the *Oracle Database Globalization Support Guide*.

71.1 Language Support in ADF Faces Components

Some ADF Faces components include text that is part of the component, for example the `af:table` component uses the resource string `af_table.LABEL_FETCHING` for the message text that is displayed in the browser while the table is fetching data during the initial load of data or while the table is being scrolled. Oracle JDeveloper provides translations of these text resources into 28 languages. Therefore, when using the supplied components, you do not need to perform additional steps to translate the text in the components.

71.2 Using Resource Bundles to Support Multiple Languages

For any text you add to a component, for example if you define the label of an `af:commandButton` component by setting the text attribute, you must provide a resource bundle that holds the actual text, create a version of the resource bundle for each locale, and add a `<locale-config>` element to define default and support locales in the application's faces-config.xml file. You must also add a `<resource-bundle>` element to your application's faces-config.xml file to make the resource bundles available to all the pages in your application. Once you have configured and registered a resource bundle, the Expression Language (EL) editor displays the key from the bundle, making it easier to reference the bundle in application pages.

To simplify the process of creating text resources for text you add to ADF components, JDeveloper supports automatic resource bundle synchronization for any translatable string in the visual editor. When you edit components directly in the visual editor or in the Property Inspector, text resources are automatically created in the base resource bundle.

For more information on automatic resource bundles, see "Using Automatic Resource Bundle Integration in JDeveloper" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

For more information on manually defining resource bundles, see "Manually Defining Resource Bundles and Locales" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

Part XIII

Appendixes

Part XIII contains the following appendixes:

- Appendix A, "Files for WebCenter Portal: Framework Applications"
- Appendix B, "Composer Component Properties and Files"
- Appendix C, "Resource Catalog Properties and Files"
- Appendix D, "Guidelines for Creating Task Flows to Be Used in Composer-Enabled Pages"
- Appendix E, "Additional Portlet Configuration"
- Appendix F, "Reuse of Oracle Portal Components"
- Appendix G, "Expression Language Expressions"
- Appendix H, "Manually Migrating a Framework Application"
- Appendix I, "WebCenter Portal Analytics Database Schema,"

A

Files for WebCenter Portal: Framework Applications

This appendix provides reference information about the files that are created and modified as you build up your WebCenter Portal: Framework application. This appendix includes the following sections:

- Section A.1, "About Files"
- Section A.2, "Files Overview"
- Section A.3, "Files Related to WebCenter Portal: Framework Applications"
- Section A.4, "Files Related to JSR 286 Portlets"
- Section A.5, "Files Related to PDK-Java Portlets"
- Section A.6, "System Files"
- Section A.7, "Files Related to Security"
- Section A.8, "Oracle JDeveloper Files"
- Section A.9, "Files Related to Oracle WebCenter Portal Services"

For a complete reference for the Oracle Application Development Framework (Oracle ADF) metadata files that you create in your data model and user interface projects, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

To learn more how files are affected by major actions, see Section 4.2.2, "Developer Actions Affecting Metadata Files."

A.1 About Files

When you use Oracle WebCenter Framework to build applications and components, several files are created as you perform such actions as building and consuming portlets. As you work with your application, you may find it useful to know a little bit about each of these files and how they relate to your application. You can group the files affected by the Oracle WebCenter Framework into two broad categories as follows:

- Files that are common to any Oracle ADF application, such as `web.xml`. For these files, it is useful to know what specific additions and changes are made for Framework applications. Those modifications are described in this appendix, but for more complete descriptions of these common files, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

- Files that are unique to Framework applications, such as `portlet.xml`. For these files, it is useful to know what the file is for and what it contains. These files are described completely in this appendix.

A.2 Files Overview

The files that are created and modified are closely associated with the objects that you create as part of your Framework application. Hence, the easiest way to discuss these files is by object as follows:

- Files Related to JSR 286 Portlets
- Files Related to PDK-Java Portlets
- System Files
- Files Related to Security
- Oracle JDeveloper Files
- Files Related to Oracle WebCenter Portal Services

A.3 Files Related to WebCenter Portal: Framework Applications

This section includes the following subsections that describe the files that are created for you when you create an application using WebCenter Portal's Framework application template:

- Section A.3.1, "catalog-registry.xml"
- Section A.3.2, "default-catalog.xml"
- Section A.3.3, "default-navigation-model.xml"
- Section A.3.4, "index.html"
- Section A.3.5, "navigation-registry.xml"
- Section A.3.6, "pages.xml"
- Section A.3.7, "error.jspx"
- Section A.3.8, "home.jspx"
- Section A.3.9, "login.jspx"
- Section A.3.11, "pageTemplate_globe.jspx"
- Section A.3.12, "pageTemplate_swooshy.jspx"

There are also image files created that are used in the application.

A.3.1 catalog-registry.xml

`catalog-registry.xml` is a catalog registry file. It is used by the Resource Manager when a user creates or edits a resource catalog. The registry defines the superset of all items that are available to be included in a catalog that you create or edit. You can create multiple catalogs for an application, but an application can only have one catalog registry file.

A.3.2 default-catalog.xml

`default-catalog.xml` is a default resource catalog. When a user edits pages at runtime using Composer, the default catalog file specifies all of the items that are available to be added to a page.

A.3.3 default-navigation-model.xml

`default-navigation-model.xml` is the navigation model file, which incorporates navigation structures in portal site. For more information, see Chapter 11, "Building a Navigation Model for Your Portal" and Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry."

A.3.4 index.html

`index.html` contains a redirect statement that refers to the portal navigation model, including an URL element, `pages_home`, which refers to the "home" page specified in the page hierarchy. For more information, see Section 9.13, "Changing the Default Home Page and Login/Logout Target Pages"

A.3.5 navigation-registry.xml

`navigation-registry.xml` is used by the Resource Manager when a user creates or edits a navigation model. The registry defines the superset of all items that are available to be included in a model that you create or edit. You can create multiple navigation models for an application, but an application can only have one navigation registry file. For more information, see Chapter 11, "Building a Navigation Model for Your Portal."

A.3.6 pages.xml

`pages.xml`, or the "page hierarchy" file, organizes pages into a tree structure, with a parent-child relationship between pages. This hierarchical structure allows the convenient propagation or inheritance of security settings from pages to sub pages. For more information, see Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

A.3.7 error.jspx

`error.jspx` is the default error page displayed to users when WebCenter Portal encounters an error.

A.3.8 home.jspx

`home.jspx` is the default Home page, providing users access to features such as the Documents service, Links service, Tags service, and Discussion service.

A.3.9 login.jspx

`login.jspx` is the default login page that provides login/logout functionality to users. For information on customizing the login page, see Section 69.9, "Adding Portlets to a Login Page."

A.3.10 navigation-renderer.jspx

`navigation-renderer.jspx` is responsible for rendering non-page resources that appear in a navigation, such as portlets, content, and task flows. For more information, see Section 11.5, "Editing the Navigation Renderer."

A.3.11 pageTemplate_globe.jspx

`pageTemplate_globe.jspx` is one of two seeded page templates created when you select the **Configure the application with standard Portal features** option when creating an application. This template offers essentially the same functionality as `pageTemplate_swooshy.jxpx` but with different graphics. For more information, see Section 12.2, "Working with Page Templates."

A.3.12 pageTemplate_swooshy.jspx

`pageTemplate_swooshy.jspx` is one of two seeded page templates created when you select the **Configure the application with standard Portal features** option when creating an application. This template offers essentially the same functionality as `pageTemplate_globe.jxpx` but with different graphics. For more information, see Section 12.2, "Working with Page Templates."

A.4 Files Related to JSR 286 Portlets

This section includes the following subsections that describe the files that are created for you when you build a JSR 286 portlet:

- Section A.4.1, "portlet.xml"
- Section A.4.2, "oracle-portlet-tags.jar"
- Section A.4.3, "portlet_mode.jsp"
- Section A.4.4, "portlet_name.java"
- Section A.4.5, "portlet_nameBundle.jar"
- Section A.4.6, "web.xml"

A.4.1 portlet.xml

`portlet.xml` defines the characteristics of your JSR 286 portlet. For complete details on `portlet.xml`, you should see the Java Portlet Specification available at:

`http://jcp.org/en/jsr/detail?id=168`

Example A–1 provides a sample fragment from a `portlet.xml` file. Note that this example does not include all of the available elements of `portlet.xml`.

Example A–1 portlet.xml Sample

```
<portlet>
   <description xml:lang="en">JSR 286 map portlet </description>
   <portlet-name>portlet1</portlet-name>
   <display-name xml:lang="en">Map Portlet</display-name>
   <portlet-class>jsrportlet.MapPortlet</portlet-class>
   <expiration-cache>0</expiration-cache>
   <supports>
      <mime-type>text/html</mime-type>
      <portlet-mode>edit</portlet-mode>
      <portlet-mode>help</portlet-mode>
```

```
      <portlet-mode>about</portlet-mode>
   </supports>
   <supported-locale>en</supported-locale>
   <resource-bundle>jsrportlet.resource.MapPortletBundle</resource-bundle>
   <portlet-info>
      <title>Map Portlet</title>
      <short-title>Map</short-title>
      <keywords/>
   <portlet-preferences>
      <preference>
         <name>portletTitle</name>
      </preference>
   </portlet-preferences>
   <security-role-ref>
      <role-name>viewer</role-name>
   </security-role-ref>
</portlet>
```

For JSR 286 portlets, the `portlet.xml` file contains all information related to portlets and their settings. Note that not all of these settings are used in the previous sample.

- `<description>` describes the portlet, providing details to the end user.
- `<portlet-name>` uniquely identifies the portlet within the Portlet Producer application.
- `<display-name>` is used when presenting a list of available portlets to the user.
- `<portlet-class>` contains the fully qualified class name of the class implementing the `javax.portlet.Portlet` interface or extending the `GenericPortlet` abstract class that becomes the entry point for the portlet logic. The portlet container uses this class when it invokes the portlet life cycle methods. For JSF portlets created using the Oracle JSF Portlet Bridge, this is `oracle.portlet.bridge.adf.application.ADFBridgePortlet`.
- `<expiration-cache>` the default duration (in seconds) of the expiration cache.
- `<init-param>` defines initialization parameters for configuring the behavior of the portlet. The Oracle JSF Portlet Bridge uses initialization parameters to identify the entry points for the different portlet modes supported by the portlet, for example, for View mode:

  ```
  <init-param>
    <name>javax.portlet.faces.defaultViewId.view</name>
    <value>/myPage.jspx</value>
  </init-param>
  ```

  Initialization parameters for other WebCenter Portal-supported modes are:

  - Edit mode: `javax.portlet.faces.defaultViewId.edit`
  - Help mode: `javax.portlet.faces.defaultViewId.help`
  - About mode: `javax.portlet.faces.defaultViewId.about`
  - Config mode: `javax.portlet.faces.defaultViewId.config`
  - Edit Defaults mode: `javax.portlet.faces.defaultViewId.edit_defaults`
  - Preview mode: `javax.portlet.faces.defaultViewId.preview`
  - Print mode: `javax.portlet.faces.defaultViewId.print`

Note: The value for the `defaultViewId` is relative to the application context root and must always start with a `/`. In the example provided, the value for `defaultViewId.view` is `/myPage.jspx`.

If you add a `defaultViewId` for other portlet modes, then you must also add the mode to the `<supports>` tag. For example, `<portlet-mode>HELP</portlet-mode>`.

- `<supports>` provides information about the portlet modes supported for each content type. Portlet modes supported by WebCenter Portal include: `edit`, `help`, `about`, `config`, `edit_defaults`, `preview`, and `print`.
- `<supported-locale>` lists the locales supported by the portlet at runtime.
- `<resource-bundle>` is the fully qualified class name of the resource bundle used to provide language specific portlet information, such as title and keywords.
- `<title>` is the static title of the portlet, usually displayed in the portlet decoration on the portlet window.
- `<short-title>` is the title that is used on devices (such as mobile phones) that have limited display capabilities.
- `<keywords>` are used by applications that offer search capabilities for their users.
- `<portlet-preferences>` are preference attribute definitions.
- `<supported-processing-event>` are events that the portlet can receive.
- `<supported-publishing-event>` are events that the portlet raises.
- `<supported-public-render-parameter>` is a public render parameter supported by the portlet.
- `<container-runtime-option>` an option for defining additional runtime behavior.
- `<security-role-ref>` maps a role name to a security role in `web.xml`. The list of roles in `web.xml` that the `<security-role-ref>` maps to is published to the consumer as the producer's user categories. In `web.xml`, `<security-role>` appears similar to the following:

```
<security-role>
   <description>Viewer role</description>
   <role-name>viewer</role-name>
</security-role>
```

A.4.2 oracle-portlet-tags.jar

`oracle-portlet-tags.jar` is the Oracle implementation of the JSP tag library defined by the Java Portlet Specification.

A.4.3 portlet_mode.jsp

Depending on the implementation style of the portlet mode that you choose to create for your portlet, a corresponding JSP file is created in your *`portlet_name`*`\html` directory to define that mode. For example, if you choose to have View and Edit modes for your portlet, then you need `view.jsp` and `edit.jsp` in your *`portlet_name`*`\html` directory. For JSR 286 portlets, you can have the following JSP files for your portlet modes:

- about.jsp
- config.jsp
- edit_defaults.jsp
- edit.jsp
- help.jsp
- preview.jsp
- print.jsp
- view.jsp

For further explanation of portlet modes, see Section 61.1.1, "Guidelines for Portlet Modes."

A.4.4 portlet_name.java

portlet_name.java is the class that acts as the entry point for the portlet logic. This class must implement the javax.portlet.Portlet interface or extend the GenericPortlet abstract class. The portlet container uses this class when it invokes the portlet lifecycle methods.

A.4.5 portlet_nameBundle.jar

*portlet_name*Bundle.jar is a resource bundle class, containing translation of the strings used by the portlet.

A.4.6 web.xml

web.xml is a Java EE standard descriptor that contains details about Web applications. For more information about web.xml, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

A.5 Files Related to PDK-Java Portlets

This section includes the following subsections that describe the files that are created for you when you build a PDK-Java portlet:

- Section A.5.2, "_default.properties"
- Section A.5.3, "index.jsp"
- Section A.5.4, "portlet_name_modePage.jsp"
- Section A.5.1, "producer_name.properties"
- Section A.5.5, "provider.xml"
- Section A.5.6, "web.xml"

A.5.1 producer_name.properties

producer_name.properties specifies deployment details about the producer, such as the location of the provider.xml file. For example, this file is used if the registration URL to the PDK-Java samples is of the form:

http://*host*:*port*/jpdk/provider/samples

or:

`http://host:port/jpdk/provider`

where the service ID field contains `samples`. See also Section A.5.2, "_default.properties".

A.5.2 _default.properties

`_default.properties` specifies deployment details about the producer, such as the location of the `provider.xml` file. For example, this file is used if the registration URL to the PDK-Java samples is of the form:

`http://host:port/jpdk/provider`

Note that the producer name is not supplied here so it has to default. See also Section A.5.1, "producer_name.properties".

A.5.3 index.jsp

`index.jsp` serves as a convenient starting point when testing PDK-Java producers from Oracle JDeveloper. This file lists all of the producers available in the application.

A.5.4 portlet_name_modePage.jsp

Depending on the implementation style of the portlet mode that you choose to create for your portlet, a corresponding JSP file is created in your `\htdocs\portlet_name` directory to define that mode. For example, if you choose to have View and Edit modes for a portlet named `portletOne`, then you need `portletOneShowPage.jsp` and `PortletOneEditPage.jsp` in your `\htdocs\portletOne` directory. For PDK-Java portlets, you can have the following JSP files for your portlet modes:

- `portlet_nameAboutPage.jsp`
- `portlet_nameEditDefaultsPage.jsp`
- `portlet_nameEditPage.jsp`
- `portlet_nameHelpPage.jsp`
- `portlet_nameShowDetailsPage.jsp`
- `portlet_nameShowPage.jsp`

For further explanation of portlet modes, see Section 61.1.1, "Guidelines for Portlet Modes".

A.5.5 provider.xml

`provider.xml` is the definition file for your PDK-Java producer.

> **Note:** PDK-Java producers were formerly referred to as providers.

This section includes the following subsections:

- Section A.5.5.1, "provider.xml Syntax"
- Section A.5.5.2, "provider.xml Sample"

A.5.5.1 provider.xml Syntax

For more information about the elements and syntax of `provider.xml`, see the Provider Definition Extensible Markup Language (XML) Tag Reference v2 on the Oracle Technology Network.

http://www.oracle.com/technology/products/ias/portal/html/javadoc/xml_tag_reference_v2.html

A.5.5.2 provider.xml Sample

Example A–2 provides a sample `provider.xml` file.

Example A–2 provider.xml Sample

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<?providerDefinition version="3.1"?>
<provider class="oracle.portal.provider.v2.DefaultProviderDefinition">
 <localePersonalizationLevel>none</localePersonalizationLevel>
 <session>true</session>
 <defaultLocale>en</defaultLocale>
 <preferenceStore
  class="oracle.portal.provider.v2.preference.FilePreferenceStore">
  <name>prefStore1</name>
  <useHashing>true</useHashing>
 </preferenceStore>
 <portlet class="oracle.portal.provider.v2.DefaultPortletDefinition">
  <id>1</id>
  <name>SampleRenderer</name>
  <title>SampleRenderer example</title>
  <shortTitle>SampleRenderer</shortTitle>
  <description>Example portlet rendered using the SampleRenderer</description>
  <timeout>40</timeout>
  <timeoutMessage>SampleRenderer example timed out</timeoutMessage>
  <acceptContentType>text/html</acceptContentType>
  <showEdit>true</showEdit>
  <showEditToPublic>false</showEditToPublic>
  <showEditDefault>true</showEditDefault>
  <showPreview>true</showPreview>
  <showDetails>true</showDetails>
  <hasHelp>true</hasHelp>
  <hasAbout>true</hasAbout>
  <renderer
   class="oracle.portal.sample.v2.devguide.samplerenderer.SampleRenderer"/>
   <personalizationManager
   class="oracle.portal.provider.v2.personalize.PrefStorePersonalizationManager">
   <dataClass>oracle.portal.provider.v2.personalize.
     NameValuePersonalizationObject
   </dataClass>
   </personalizationManager>
 </portlet>
</provider>
```

A.5.6 web.xml

`web.xml` is a Java EE standard descriptor that contains details about Web applications. For more information about `web.xml`, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

A.6 System Files

This section includes the following subsections that describe the files that are created or modified when you create or modify pages, deploy your application, or consume JSR 268 portlets:

- Created at page design time:
  - Section A.6.1, "adf-config.xml"
  - Section A.6.2, "DataBindings.cpx"
  - Section A.6.3, "faces-config.xml"
  - Section A.6.4, "page_name.jspx"
  - Section A.6.5, "PageDef.xml"
  - Section A.6.6, "trinidad-config.xml"
  - Section A.6.7, "web.xml"
  - Section A.6.8, "weblogic-application.xml"
- Created at application deployment time:
  - Section A.6.9, "mds Subdirectory"
- Created upon consumption of JSR 286 portlets
  - Section A.6.10, "wsdl Subdirectory"

A.6.1 adf-config.xml

The `adf-config.xml` file is an extensible configuration file used by any ADF component. This file is used by the various WebCenter Portal services, including pages, portlets, and all the services such as search, worklist, forums, and so on, to set configuration information. This file, with `connections.xml`, contains the configuration information required for most of the services to function.

For more information, see:

- Section 15.10.1, "adf-config.xml" (for information on Skins settings)
- Section 21.2.1.3, "Registering Add-Ons in adf-config.xml"
- Section 22.11.2.2, "Adding Relevant Entries to the adf-config.xml File" (for information on configuring change persistence)
- Section 23.2.1.1, "Updating Your Application's adf-config.xml File" (for information on configuring sandbox creation)
- Section 24.5.1, "How to Add an enableSecurity Section to adf-config.xml"
- Section B.2.2, "adf-config.xml" (for information on Composer-specific settings)
- Section H.5.6, "Updating adf-config.xml" (for information on migration)

Table A–1 describes the child elements of `<adf-portlet-config>`. The `init-param` names in the first column correspond to the names of the servlet `init-params` used when the portlet client is accessed through the Web adapter.

Table A–1 Child Elements of adf-portlet-config

Element (init-param)	Description	Default Value
`parallelPoolSize (parallel.pool.size)`	The number of threads to use for parallel execution of tasks.	10

Table A–1 (Cont.) Child Elements of adf-portlet-config

Element (init-param)	Description	Default Value
`parallelQueueSize` (`parallel.queue.size`)	The size of the queue of tasks waiting for parallel execution. Tasks are rejected when the queue size is exceeded.	20
`defaultTimeout` (`default.timeout`)	The default timeout period in seconds for requests made to producers. This value is used when a timeout is not defined at the portlet or producer level.	10
`minimumTimeout` (`minimum.timeout`)	The minimum timeout period in seconds for requests made to producers. This value is used to impose a lower limit on timeout periods specified by portlets or producers.	0.1
`maximumTimeout` (`maximum.timeout`)	The maximum timeout period in seconds for requests made to producers. This value is used to impose an upper limit on timeout periods specified by portlets or producers.	300
`resourceProxyPath` (`resource.proxy.path`)	The base path of the resource proxy servlet, relative to the context root of the application. Used to construct links to the resource servlet within portlet markup.	/resourceproxy
`supportedLocales` (`supported.locales`)	The set of supported locales defined using strings of the form: `language[_country[_variant]]`	Commented out by default. You should uncomment it if you have multiple locales. See Example A–3.
`portletTechnologies` (`portlet.technologies`)	The set of portlet technologies supported by the client defined by the fully qualified names of classes that implement the PortletTechnologyConfig interface.	{o.p.c.ci.web.WebPortletTechnologyConfig, o.p.c.ci.wsrp.WSRPPortletTechnologyConfig}
`cacheSettings` (`cache.*`)	Cache configuration information. Used to enable or disable the cache, define its maximum size and impose limits on the amount of space available for different users and subscribers.	The cache is enabled and no size restrictions are imposed.

Table A–1 (Cont.) Child Elements of adf-portlet-config

Element (init-param)	Description	Default Value
`maximumResourceUrlLength` (`maximum.resource.url.length`)	The maximum length allowed for resource URLs. This prevents the URL from becoming to long if the portlet passes a lot of contextual information in a resource request. Resource URLs longer than 2048 characters may cause problems in some browsers. If the information encoded in a resource URL exceeds this value, the portlet client stores the information in the session and encodes a key to the session attribute in the URL instead. The key is calculated in a deterministic way to avoid adversely affecting browser caching.	2048

<adf-portlet-config> element Sample

Example A–3 illustrates the usage of the `<adf-portlet-config>` element.

Example A–3 <adf-portlet-config> element Sample

```
<adf-portlet-config xmlns="http://xmlns.oracle.com/adf/portlet/config">
  <supportedLocales>
    <value>en</value>
    <value>fr</value>
    <value>de</value>
    <value>es</value>
  </supportedLocales>
  <portletTechnologies>
   <value>oracle.portlet.client.containerimpl.web.
     WebPortletTechnologyConfig</value>
   <value>oracle.portlet.client.containerimpl.wsrp.
     WSRPPortletTechnologyConfig</value>
  </portletTechnologies>
  <defaultTimeout>20</defaultTimeout>
  <minimumTimeout>1</minimumTimeout>
  <maximumTimeout>300</maximumTimeout>
  <resourceProxyPath>/portletresource</resourceProxyPath>
  <cacheSettings>
    <maxSize>10000000</maxSize>
    <subscriber default="true">
      <systemLevel>
        <maxSize>5000000</maxSize>
      </systemLevel>
      <userLevel>
        <maxSize>8000000</maxSize>
      </userLevel>
    </subscriber>
  </cacheSettings>
</adf-portlet-config>
```

A.6.2 DataBindings.cpx

`DataBindings.cpx` is a file common to Web applications. For more information about `DataBindings.cpx`, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

A.6.3 faces-config.xml

`faces-config.xml` is a file common to JSF applications. It describes the page flow of your application. For more information about `faces-config.xml`, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

A.6.4 page_name.jspx

`page_name``.jspx` is the JSP file for your page. Whenever you add or remove components, such as portlets or data controls from the page, this file is updated.

A.6.5 PageDef.xml

`PageDef.xml` is a file common to Oracle ADF applications. This file holds information about portlet bindings. Also, portlet parameters can be tied to page variables in this file.

For more information about `PageDef.xml`, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

PageDef.xml Sample

Example A–4 provides a sample `PageDef.xml` file.

Example A–4 PageDef.xml Sample

```
<?xml version="1.0" encoding="UTF-8" ?>
<pageDefinition xmlns="http://xmlns.oracle.com/adfm/uimodel"
   version="10.1.3.37.97" id="app_SRFeedbackPageDef"
   Package="oracle.srdemo.view.pageDefs">
  <parameters/>
  <executables>
    <methodIterator id="findAllServiceRequestIter"
                    Binds="findAllServiceRequest.result"
                    DataControl="SRPublicFacade" RangeSize="4"
                    BeanClass="oracle.srdemo.model.entities.ServiceRequest"/>
    <variableIterator id="variables">
      <variable Name="portlet1_Param1" Type="java.lang.Object"
      DefaultValue="
      ${bindings.findAllServiceRequestIter.currentRow.dataProvider['svrId']}"/>
      <variable Name="portlet1_Param2" Type="java.lang.Object"/>
      <variable Name="portlet1_Param3" Type="java.lang.Object"/>
      <variable Name="portlet1_Param4" Type="java.lang.Object"/>
      <variable Name="portlet1_Param5" Type="java.lang.Object"/>
    </variableIterator>
    <methodIterator id="findServiceRequestByIdIter"
                    Binds="findServiceRequestById.result"
                    DataControl="SRPublicFacade" RangeSize="10"
                    BeanClass="oracle.srdemo.model.entities.ServiceRequest"/>
    <portlet id="portlet1"
      portletInstance="/oracle/adf/portlet/OmniProducer_1150310748178/
      applicationPortlets/Portlet100_eebc7f18_010b_1000_8001_82235f640cea"
      class="oracle.adf.model.portlet.binding.PortletBinding"
```

```
      xmlns="http://xmlns.oracle.com/portlet/bindings">
    <parameters>
      <parameter name="Param1" pageVariable="portlet1_Param1"/>
      <parameter name="Param2" pageVariable="portlet1_Param2"/>
      <parameter name="Param3" pageVariable="portlet1_Param3"/>
      <parameter name="Param4" pageVariable="portlet1_Param4"/>
      <parameter name="Param5" pageVariable="portlet1_Param5"/>
    </parameters>
  </portlet>
</executables>
<bindings>
  <methodAction id="findAllServiceRequest"
                InstanceName="SRPublicFacade.dataProvider"
                DataControl="SRPublicFacade"
                MethodName="findAllServiceRequest" RequiresUpdateModel="true"
                Action="999" IsViewObjectMethod="false"
                ReturnName="SRPublicFacade.methodResults.
                   SRPublicFacade_dataProvider_findAllServiceRequest_result"/>
  <table id="findAllServiceRequest1" IterBinding="findAllServiceRequestIter">
    <AttrNames>
      <Item Value="assignedDate"/>
      <Item Value="problemDescription"/>
      <Item Value="requestDate"/>
      <Item Value="status"/>
      <Item Value="svrId"/>
      <Item Value="custComment"/>
      <Item Value="custCommentDate"/>
      <Item Value="custCommentContactBy"/>
      <Item Value="mgrNotes"/>
      <Item Value="mgrNotesDate"/>
    </AttrNames>
  </table>
  <methodAction id="findServiceRequestById"
                InstanceName="SRPublicFacade.dataProvider"
                DataControl="SRPublicFacade"
                MethodName="findServiceRequestById" RequiresUpdateModel="true"
                Action="999" IsViewObjectMethod="false"
                ReturnName="SRPublicFacade.methodResults.
                   SRPublicFacade_dataProvider_findServiceRequestById_result">
    <NamedData NDName="svrIdParam"
       NDValue=
        "${bindings.findAllServiceRequestIter.currentRow.dataProvider['svrId']}"
       NDType="java.lang.Integer"/>
  </methodAction>
  <attributeValues id="svrId" IterBinding="findServiceRequestByIdIter">
    <AttrNames>
      <Item Value="svrId"/>
    </AttrNames>
  </attributeValues>
  <attributeValues id="custComment" IterBinding="findServiceRequestByIdIter">
    <AttrNames>
      <Item Value="custComment"/>
    </AttrNames>
  </attributeValues>
  <attributeValues id="mgrNotes" IterBinding="findServiceRequestByIdIter">
    <AttrNames>
      <Item Value="mgrNotes"/>
    </AttrNames>
  </attributeValues>
  <list id="ServiceRequestcustCommentContactBy"
```

```
            IterBinding="findServiceRequestByIdIter" ListOperMode="0"
            StaticList="true" NullValueFlag="1">
          <AttrNames>
            <Item Value="custCommentContactBy"/>
          </AttrNames>
          <ValueList>
            <Item Value=" "/>
            <Item Value="Phone"/>
            <Item Value="Email"/>
            <Item Value="SMS"/>
          </ValueList>
        </list>
        <methodAction id="mergeEntity" InstanceName="SRPublicFacade.dataProvider"
                      DataControl="SRPublicFacade" MethodName="mergeEntity"
                      RequiresUpdateModel="true" Action="999"
                      IsViewObjectMethod="false"
                      ReturnName="SRPublicFacade.methodResults.
                                  SRPublicFacade_dataProvider_mergeEntity_result">
          <NamedData NDName="entity"
             NDValue=
             "${bindings.findServiceRequestByIdIter.currentRow.dataProvider}"
             NDType="java.lang.Object"/>
        </methodAction>
      </bindings>
    </pageDefinition>
```

A.6.6 trinidad-config.xml

`trinidad-config.xml` contains information about the application skin family. Additionally, it can contain information about the level of page accessibility support, page animation, time zone, enhanced debugging output, and the URL for Oracle Help for the Web (OHW). For more information, see Section 1.10.3, "The Framework Application Template Default trinidad-config.xml File"

A.6.7 web.xml

`web.xml` is a Java EE standard descriptor that contains details about Web applications. For more information about `web.xml`, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

A.6.8 weblogic-application.xml

`weblogic-application.xml` is a deployment descriptor (a server configuration file that defines the application configuration) for Oracle WebLogic Server deployment.

A.6.9 mds Subdirectory

When you deploy a WebCenter Portal: Framework application, an `mds` subdirectory is created in your project directory. This subdirectory contains other subdirectories and metadata files, such as portlet customizations and personalizations.

A.6.10 wsdl Subdirectory

When you consume JSR 286 portlets in an application, a `wsdl` subdirectory is created in the `WEB-INF` directory. The files in this subdirectory are internal files created for portlets from a WSRP producer.

A.7 Files Related to Security

This section includes the following subsections that describe the files that are created or modified when you implement or modify security:

- Section A.7.1, "connections.xml"
- Section A.7.2, "jazn-data.xml"
- Section A.7.3, "cwallet.sso"
- Section A.7.4, "jps-config.xml"
- Section A.7.5, "adf-config.xml"

A.7.1 connections.xml

`connections.xml` contains portlet producer connection information. Example A–5 shows a sample `connections.xml` file.

Example A–5 connections.xml

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<References xmlns="http://xmlns.oracle.com/adf/jndi">
  <Reference name="SampleWSRPProducer-wsconn"

className="oracle.adf.model.connection.webservice.impl.WebServiceConnectionImpl"
xmlns="">
     <Factory
className="oracle.adf.model.connection.webservice.api.WebServiceConnectionFactory"
/>
     <RefAddresses>
        <XmlRefAddr addrType="WebServiceConnection">
           <Contents>
              <wsconnection
description="http://portlet.example.com:9999/portletapp/portlets/wsrp2?WSDL">
                 <model name="{urn:oasis:names:tc:wsrp:v2:wsdl}WSRP_v2_Service"
xmlns="http://oracle.com/ws/model">
                    <service name="{urn:oasis:names:tc:wsrp:v2:wsdl}WSRP_v2_
Service">
                       <port name="WSRP_v2_PortletManagement_Service"
binding="{urn:oasis:names:tc:wsrp:v2:bind}WSRP_v2_PortletManagement_Binding_SOAP">
                          <soap
addressUrl="http://host:port/portletapp/portlets/WSRP_v2_PortletManagement_
Service"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                          <operation name="setPortletProperties">
                             <soap
soapAction="urn:oasis:names:tc:wsrp:v2:setPortletProperties"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                             <input name="setPortletProperties"/>
                             <output name="setPortletPropertiesResponse"/>
                          </operation>
                          <operation name="getPortletProperties">
                             <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getPortletProperties"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                             <input name="getPortletProperties"/>
                             <output name="getPortletPropertiesResponse"/>
                          </operation>
                          <operation name="importPortlets">
                             <soap
```

```
soapAction="urn:oasis:names:tc:wsrp:v2:importPortlet"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="importPortletsResponse"/>
                              <input name="importPortlets"/>
                           </operation>
                           <operation name="destroyPortlets">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:destroyPortlets"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="destroyPortlets"/>
                              <output name="destroyPortletsResponse"/>
                           </operation>
                           <operation name="exportPortlets">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:exportPortlet"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="exportPortletsResponse"/>
                              <input name="exportPortlets"/>
                           </operation>
                           <operation name="releaseExport">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:importPortlet"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="releaseExport"/>
                              <output name="releaseExportResponse"/>
                           </operation>
                           <operation name="getPortletsLifetime">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getPortletsLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="getPortletsLifetimeResponse"/>
                              <input name="getPortletsLifetime"/>
                           </operation>
                           <operation name="copyPortlets">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:copyPortlets"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="copyPortletsResponse"/>
                              <input name="copyPortlets"/>
                           </operation>
                           <operation name="getPortletPropertyDescription">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getPortletPropertyDescription"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="getPortletPropertyDescription"/>
                              <output
name="getPortletPropertyDescriptionResponse"/>
                           </operation>
                           <operation name="clonePortlet">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:clonePortlet"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="clonePortletResponse"/>
                              <input name="clonePortlet"/>
                           </operation>
                           <operation name="setPortletsLifetime">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:setPortletsLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="setPortletsLifetimeResponse"/>
```

```
                           <input name="setPortletsLifetime"/>
                        </operation>
                        <operation name="getPortletDescription">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getPortletDescription"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <input name="getPortletDescription"/>
                           <output name="getPortletDescriptionResponse"/>
                        </operation>
                        <operation name="setExportLifetime">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:setExportLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <input name="setExportLifetime"/>
                           <output name="setExportLifetimeResponse"/>
                        </operation>
                     </port>
                     <port name="WSRP_v2_Markup_Service"
binding="{urn:oasis:names:tc:wsrp:v2:bind}WSRP_v2_Markup_Binding_SOAP">
                        <soap
addressUrl="http://host:port/portletapp/portlets/WSRP_v2_Markup_Service"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                        <operation name="initCookie">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:initCookie"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <output name="initCookieResponse"/>
                           <input name="initCookie"/>
                        </operation>
                        <operation name="getMarkup">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getMarkup"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <output name="getMarkupResponse"/>
                           <input name="getMarkup"/>
                        </operation>
                        <operation name="releaseSessions">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:releaseSessions"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <input name="releaseSessions"/>
                           <output name="releaseSessionsResponse"/>
                        </operation>
                        <operation name="performBlockingInteraction">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:performBlockingInteraction"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <input name="performBlockingInteraction"/>
                           <output name="performBlockingInteractionResponse"/>
                        </operation>
                        <operation name="getResource">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getResource"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <output name="getResourceResponse"/>
                           <input name="getResource"/>
                        </operation>
                        <operation name="handleEvents">
                           <soap
soapAction="urn:oasis:names:tc:wsrp:v2:handleEvents"
```

```
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="handleEvents"/>
                              <output name="handleEventsResponse"/>
                           </operation>
                        </port>
                        <port name="WSRP_v2_Registration_Service"
binding="{urn:oasis:names:tc:wsrp:v2:bind}WSRP_v2_Registration_Binding_SOAP">
                           <soap
addressUrl="http://host:port/portletapp/portlets/WSRP_v2_Registration_Service"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <operation name="register">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:register"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="register"/>
                              <output name="registerResponse"/>
                           </operation>
                           <operation name="getRegistrationLifetime">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getRegistrationLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <output name="getRegistrationLifetimeResponse"/>
                              <input name="getRegistrationLifetime"/>
                           </operation>
                           <operation name="setRegistrationLifetime">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:setRegistrationLifetime"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="setRegistrationLifetime"/>
                              <output name="setRegistrationLifetimeResponse"/>
                           </operation>
                           <operation name="deregister">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:deregister"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="deregister"/>
                              <output name="deregisterResponse"/>
                           </operation>
                           <operation name="modifyRegistration">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:modifyRegistration"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="modifyRegistration"/>
                              <output name="modifyRegistrationResponse"/>
                           </operation>
                        </port>
                        <port name="WSRP_v2_ServiceDescription_Service"
binding="{urn:oasis:names:tc:wsrp:v2:bind}WSRP_v2_ServiceDescription_Binding_
SOAP">
                           <soap
addressUrl="http://host:port/portletapp/portlets/WSRP_v2_ServiceDescription_
Service"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                           <operation name="getServiceDescription">
                              <soap
soapAction="urn:oasis:names:tc:wsrp:v2:getServiceDescription"
xmlns="http://schemas.xmlsoap.org/wsdl/soap/"/>
                              <input name="getServiceDescription"/>
                              <output name="getServiceDescriptionResponse"/>
                           </operation>
```

```
                    </port>
                  </service>
                </model>
              </wsconnection>
            </Contents>
          </XmlRefAddr>
        </RefAddresses>
  </Reference>
  <Reference name="SampleWSRPProducer"
className="oracle.portlet.client.connection.wsrp.WSRPProducerConnection" xmlns="">
     <Factory
className="oracle.portlet.client.connection.wsrp.WSRPProducerConnectionFactory"/>
     <RefAddresses>
        <XmlRefAddr addrType="connectionDesc">
           <Contents>
              <wsrpproducerconnection wsConnection="SampleWSRPProducer-wsconn"
timeout="30"/>
           </Contents>
        </XmlRefAddr>
     </RefAddresses>
  </Reference>
</References>
```

A.7.2 jazn-data.xml

When you implement or modify security for your application, the `jazn-data.xml` file is created or modified.

The `jazn-data.xml` file is used to facilitate the deployment of realm and policy information for your application. In your development environment (JDeveloper), `jazn-data.xml` is located in your application's *`workspacedir`*`/src/META-INF` directory. After deployment, the contents of this file are merged in domain-level identity or policy stores. This file contains the policies that are created at design time, and is merged at deployment by `JpsApplicationLifecycleListener` into the `system-jazn-data.xml` file (or appropriately configured policy store) for the deployment target.

When you migrate security information with `JpsApplicationLifecycleListener`, the `jazn-data.xml` file can be used as the source file for the migration. For more information about migrating roles, see the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

jazn-data.xml Sample

Example A–6 provides a sample `jazn-data.xml` file.

Example A–6 jazn-data.xml Sample

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<jazn-data>
   <jazn-realm>
      <realm>
         <name>jazn.com</name>
      </realm>
   </jazn-realm>
   <policy-store>
      <applications>
         <application>
```

```
            <name>mdsapp-secure</name>
            <app-roles>
               <app-role>
                  <name>unauthenticated-role</name>
                  <display-name>anonymous-role</display-name>

<class>oracle.security.jps.service.policystore.ApplicationRole</class>
               </app-role>
               <app-role>
                  <name>authenticated-role</name>
                  <display-name>authenticated-users</display-name>

<class>oracle.security.jps.service.policystore.ApplicationRole</class>
               </app-role>
               <app-role>
                  <name>approle</name>
                  <display-name>approle</display-name>
                  <description>approle</description>
                  <guid/>

<class>oracle.security.jps.service.policystore.ApplicationRole</class>
                  <members>
                     <member>

<class>oracle.security.jps.internal.core.principals.JpsXmlUserImpl</class>
                        <name>orcladmin</name>
                     </member>
                  </members>

               </app-role>
            </app-roles>
            <jazn-policy>
<grant>
<grantee>
 <principals>
     <principal>
<class>oracle.security.jps.internal.core.principals.JpsAuthenticatedRoleImpl</clas
s>
<name>authenticated-role</name>
     </principal>
 </principals>
</grantee>
<permissions>
 <permission>
<class>oracle.adf.share.security.authorization.RegionPermission</class>
<name>project1.pageDefs.securePageDef</name>
<actions>view</actions>
 </permission>
</permissions>
</grant>
<grant>
<grantee>
 <principals>
     <principal>
<class>oracle.security.jps.internal.core.principals.JpsAnonymousRoleImpl</class>
         <name>anonymous-role</name>
     </principal>
 </principals>
</grantee>
<permissions>
```

```
 <permission>
<class>oracle.adf.share.security.authorization.RegionPermission</class>
<name>project1.pageDefs.publicPageDef</name>
<actions>view</actions>
 </permission>
</permissions>
</grant>
            </jazn-policy>
          </application>
       </applications>
   </policy-store>
   <jazn-policy/>
</jazn-data>
```

A.7.3 cwallet.sso

`cwallet.sso` contains external application credentials, if any, and the connection's secure data such as database password. The `cwallet.sso` file is located in the *`workspaceroot`*`/src/META-INF` directory.

A.7.4 jps-config.xml

`jps-config.xml` is located at *`workspaceroot`*`/src/META-INF`. The `jps-config.xml` file packaged with the application is used only at design time. At runtime, the configurations stored in the domain-level `jps-config.xml` file become functional.

`jps-config.xml` configures OPSS services. For more information on the element hierarchy and attributes of `jps-config.xml`, see the appendix "OPSS Configuration File Reference" in the *Oracle Containers for J2EE Security Guide*.

The recommendation for migration is to create a new `jps-config.xml` file with the source and destination contexts defined as required by the migration commands. For details on migrating identities, see the section "Migrating Identities Manually" in the *Oracle Containers for J2EE Security Guide*.

The policy store is migrated in the following ways: entire policy store, application-specific policies, and global polices. For details, see the section "Migrating Policies Manually" in the *Oracle Containers for J2EE Security Guide*.

The credential store is migrated in the following ways: entire credential store or application-specific credentials. When migrating application-specific credentials, ensure that `srcFolder` and `targetFolder` are the same. `srcFolder` and `targetFolder` can be determined from `appUID` in the `adf-config.xml` file, see Example A–7. For details, see the section "Migrating Credentials Manually" in the *Oracle Containers for J2EE Security Guide*.

Example A–7 appUID in the adf-config.xml file

```
<adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
  <adf-property name="adfAppUID" value="Application1-1167"/>
</adf:adf-properties-child>
```

A.7.5 adf-config.xml

Security configurations are stored in `adf-config.xml`. This file is located at *`workspaceroot`*`/.adf/META-INF/adf-config.xml`. See Example A–8.

Example A–8 adf-config.xml

```
<adf-security-child xmlns="http://xmlns.oracle.com/adf/security/config">
        <JaasSecurityContext
initialContextFactoryClass="oracle.adf.share.security.JAASInitialContextFactory"
jaasProviderClass="oracle.adf.share.security.providers.jps.JpsSecurityContext"
             authorizationEnforce="true"
             authenticationRequire="true"/>
    <CredentialStoreContext
credentialStoreClass="oracle.adf.share.security.providers.jps.CSFCredentialStore"
             credentialStoreLocation="../../src/META-INF/jps-config.xml"/>
</adf-security-child>
```

Note: The `credentialStoreLocation` entry in the sample is valid at design time only.

A.8 Oracle JDeveloper Files

This section includes the following subsection that describes the file that is created or modified when you create a page template:

- Section A.8.1, "pagetemplate-metadata.xml"

A.8.1 pagetemplate-metadata.xml

`pagetemplate-metadata.xml` keeps track of all the page templates you create in a project.

A.9 Files Related to Oracle WebCenter Portal Services

This section includes sample `connections.xml` and `adf-config.xml` files for some WebCenter Portal services.

Example A–9 shows the minimum required configuration for the SES Web service connection.

Example A–9 SES Web Service Sample Connection

```
<Reference name="GenericSesConnection"
className="oracle.adf.mbean.share.connection.webcenter.search.SesConnectionReferen
ceable"
credentialStoreKey="GenericSesConnection" xmlns="">
      <Factory
className="oracle.adf.mbean.share.connection.webcenter.search.SesConnectionFactory
"/>
      <RefAddresses>
         <XmlRefAddr addrType="SesConfig">
            <Contents>
               <SesConfig>
                  <AttributeMap>
                     <Attribute name="date">
                        <SesAttribute>LastModifiedDate</SesAttribute>
                     </Attribute>
                     <Attribute name="person">
                        <SesAttribute>Author</SesAttribute>
                     </Attribute>
                  </AttributeMap>
                  <SoapUrl>http://host:port/search/query/OracleSearch</SoapUrl>
```

```
            </SesConfig>
         </Contents>
      </XmlRefAddr>
      <StringRefAddr addrType="appUsername">
         <Contents>wpadmin</Contents>
      </StringRefAddr>
      <SecureRefAddr addrType="appPassword"/>
   </RefAddresses>
</Reference>
```

Example A–10 shows a sample `adf-config.xml` for the Documents service.

Example A–10 ADF Configuration for the Documents Service

```
<doclibC:adf-doclib-config xmlns="http://xmlns.oracle.com/webcenter/doclib/config"
   primaryConnectionName="Oracle_Content_Server">
    <doclibC:spaces-repository
      spacesRoot="/repositoryPath"
      adminUserName="sysadmin"
      applicationName="application_name" />
  </doclibC:adf-doclib-config>
```

Example A–11 shows a sample `adf-config.xml` for the Worklist service.

Example A–11 ADF Configuration for the Worklist Service

```
  <worklistC:adf-worklist-config>
    <worklistC:ConnectionName>WebCenter-Worklist</worklistC:ConnectionName>
  </worklistC:adf-worklist-config>
```

B

Composer Component Properties and Files

This appendix provides reference information about Composer-specific tags, configuration files, and style properties. This information is useful while performing the tasks described in Chapter 20, "Enabling Runtime Editing of Pages Using Composer" and Chapter 21, "Extending Runtime Editing Capabilities Using Composer."

This appendix contains the following sections:

- Section B.1, "Composer Component Properties"
- Section B.2, "Composer-Specific Files and Configurations"
- Section B.3, "Composer Default Add-Ons and Property Panels"
- Section B.4, "Composer Help Topic IDs"
- Section B.5, "Composer Components Style-Specific Properties"
- Section B.6, "Customizable Components (HTML) Properties"

B.1 Composer Component Properties

This section is a quick reference for all Composer tags. When you add Composer components to a page, default values are assigned to certain attributes. You can change these default values and define values for the remaining attributes in JDeveloper.

For information about adding these components to the page, see Section 20.1, "Designing Editable Pages Using Composer Components."

> **Note:** The Property Inspector in JDeveloper displays certain extended metadata attributes under the Customization Attributes category. You can use these properties to provide additional metadata information. For more information, see the section "Extended Metadata Properties" in *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

This section contains the following subsections:

- Section B.1.1, "Page Customizable"
- Section B.1.2, "Change Mode Link and Change Mode Button"
- Section B.1.3, "Layout Customizable Component"
- Section B.1.4, "Panel Customizable Component"
- Section B.1.5, "Show Detail Frame Component"

- Section B.1.6, "Custom Action"
- Section B.1.7, "Show Property"

B.1.1 Page Customizable

Use the `Page Customizable` component to enable page editing at runtime. The `Page Customizable` component denotes the customizable part of a page and shows the Composer toolbar in Edit mode at runtime. When you enclose components within a `Page Customizable` component, Composer is enabled in the application page at run time, allowing the user to customize and edit the components.

In large applications, where you want to enable runtime page editing on multiple pages, you can include the `Page Customizable` component in the page template, somewhere in the top of the hierarchy. This way, all pages created using the template are editable at runtime.

Geometry Management

View mode: The `Page Customizable` component can be stretched by a parent layout component that stretches its children, for example, a `Panel Stretch Layout` or `Panel Splitter` component. If the `Page Customizable` component is stretched, it in turn stretches its child component to fill the space available to it. However, if the `Page Customizable` component is not stretched, its size is determined by the child component.

Edit mode: The `Page Customizable` component stretches its child component to fill the space available to it.

Attributes

Table B–1 describes the attributes of a `Page Customizable` component.

Table B–1 Attributes of a Page Customizable Component

Attribute	Type	Supports EL	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`.

Table B–1 (Cont.) Attributes of a Page Customizable Component

Attribute	Type	Supports EL	Description
Appearance Attributes			
sourceViewPosition	String	Yes	The position of the Source view pane displaying components in a tree structure. Valid values are `bottom`, `end`, `start`, and `top`. Default value is `top`. Alternatively, the Source Position submenu on the View menu on the Composer toolbar provides options to set this attribute at runtime. For more information, see the section, Section 18.5, "Editing Capabilities in Source View in Page Edit Mode.".
sourceViewSize	String	Yes	The height or width of the Source view pane in pixels. Use this to specify height if the `sourceViewPosition` is set to `top` or `bottom`, or to specify width if `sourceViewPosition` is set to `start` or `end`. Since the default for `sourceViewPosition` is `top`, the pane has a default height of `200` pixels. Alternatively, the resize handler on the edge of the Source view pane enables users to alter the height or width of the pane at runtime.
toolbarLayout	String	Yes	The elements to be displayed on the Composer toolbar. This attribute can take a space-separated list of elements. The built in strings the component recognizes are `message`, `stretch`, `statusindicator`, `newline`, `menu`, `addonpanels`, `stretch`, `help`, and `button`. For more information about using this attribute, see Section 21.10, "Customizing the Composer Toolbar."
view	String	Yes	The default view in which to open a page in Composer. Available options are `design` and `source`. Default value is `design`. For more information, see Section 22.7, "Configuring an Application Page to Display in Source View by Default."
designViews	String	No	The views available to the user in Composer Design view. This attribute can take a space-separated list of elements. Available options are `select`, `add-content` and `all`. Default value is `add-content`. This setting overrides the global setting `<pe:enable-design-views>` in adf-config.xml. For more information see Section 21.11, "Enabling Direct Select in Design View."

Table B–1 (Cont.) Attributes of a Page Customizable Component

Attribute	Type	Supports EL	Description
Style Attributes			
styleClass	String	Yes	The CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle`'s CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
Advanced Attributes			
binding	`oracle.adf.view.page.editor.component.PageCustomizable`	Supports only EL	An EL reference that stores the component instance on a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean.
sourceViewNodeAction	String	Yes	The name of the JPSX file to be called if you want to display custom actions against task flows in the component navigator in Composer Source view. The `sourceViewNodeAction` attribute can take the name of a JSPX file or an EL value that evaluates to a JSPX file name. For information about using this attribute, see Section 22.4, "Enabling Custom Actions that Display on Task Flows in the Component Navigator."
allowLabel	Boolean	Yes	Specifies whether label creation must be enabled in the Customization Manager. When `allowLabel` is set to `true`, a Save and Label button is displayed on the Composer toolbar. For more information, see Section 18.4.14, "Manage Application Customizations." This attribute is relevant only if you have configured sandbox support in your application.
catalog	String	Yes	The Resource Catalog to be used as the default one for the application page. Provide the catalog definition file name. For more information about using multiple Resource Catalogs, see Chapter 16, "Creating and Managing Resource Catalogs."

Table B–1 (Cont.) Attributes of a Page Customizable Component

Attribute	Type	Supports EL	Description
allowEL	Boolean	No	Specifies whether or not EL can be run in application pages. Default is true. This setting overrides the `<pe:allow-el>` setting in adf-config.xml. For more information, see Section 21.4.2, "How to Protect Expression Language."
protectEL	Boolean	No	Specifies whether or not EL in form elements is read-only. Default is false. This setting overrides the `<pe:protect-el>` setting in adf-config.xml. For more information, see Section 21.4.2, "How to Protect Expression Language."

Supported Facets

The `Page Customizable` component provides an `editor` facet, which is prepopulated with a `Page Editor Panel` component. To ensure that Composer works properly, the `editor` facet must contain the `Page Editor Panel` component.

B.1.2 Change Mode Link and Change Mode Button

Use the `Change Mode Button` or `Change Mode Link` to enable switching to Edit mode of the page at runtime.

The `Change Mode Link` and `Change Mode Button` components share a common set of attributes. Table B–2 describes the attributes of a `Change Mode Link` or `Change Mode Button` component.

Table B–2 Attributes of a Change Mode Link or Change Mode Button Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	Yes	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Za-z) or an underscore ('_ '). ■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. In a secured application, to enable the Edit link or button only for selected users based on specific criteria, you can specify an EL value for the `rendered` attribute. **Note**: In a secured application, it is recommended that you check all users' privileges and enable the Edit link or button only for privileged users. Unauthenticated users who stumble into page Edit mode can change component properties.
Behavior Attributes			

Table B–2 (Cont.) Attributes of a Change Mode Link or Change Mode Button Component

Attribute	Type	Supports EL?	Description
immediate	Boolean	Yes	Specifies whether data validation - client-side or server-side - must be skipped when events are generated by this component. When `immediate` is `true`, the command's action and ActionListeners, including the default ActionListener provided by the JavaServer Faces implementation, is executed during the Apply Request Values phase of the request processing lifecycle, rather than waiting until the Invoke Application phase. Because validation runs during Process Validators (after Apply Request Values, but before Invoke Application), setting `immediate` to `true` skips validation.
Advanced Attributes			
binding	`oracle.adf.view.page.editor.component.ChangeModeLink` or `oracle.adf.view.page.editor.component.ChangeModeButton`	Supports only EL	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean.

If you do not want to use the `Change Mode Link` or `Change Mode Button` component, you can add your own button or link component and use the `ChangeModeLink` or `ChangeModeButton` API to enable the switching of the page mode.

Disabling Runtime Editing in Your Application Pages

To disable runtime editing of pages, you can revoke Edit privileges for users, or delete the `Change Mode Link` or `Change Mode Button` component from the page so that the user cannot navigate to the Edit mode of the page.

B.1.3 Layout Customizable Component

The `Layout Customizable` component applies a predefined layout to its child components. It is a container component that enables end users to lay out its child components in several predefined ways. Use the `Layout Customizable` component only to enable changing of the page layout at runtime. If you just want container components in which to layout components, but do not want to enable runtime layout changes, then it is recommended that you use ADF Faces layout components like `Panel Stretch Layout` and `Panel Group Layout`. It is recommended that you have just one `Layout Customizable` on the page.

A `Layout Customizable` component contains a direct child `Panel Customizable` component and two facets with `Panel Customizable` components by default to enable users to add content inside them at runtime. However, you can add any components inside the `Layout Customizable` component and its facets. The content in the three `Panel Customizable` components can be placed in different ways using predefined layouts. For more information, see the section, "Predefined Layout Types".

Table B–3 describes the attributes of a `Layout Customizable` component.

Table B–3 Attributes of the Layout Customizable Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.
type	String	Yes	Specifies the layout to be used for the page. You can choose from a list of eight predefined layouts. For more information, see Table B–5. Default value is `threeColumn`. You can define customization restrictions on a `Layout Customizable` component. Specifically, if you place a restriction on the `type` attribute, then layout changer options are shown as disabled and users cannot use them to change the page layout. For more information, see Chapter 24, "Modifying Default Security Behavior of Composer Components."
Appearance Attributes			
text	String	Yes	Specifies the text to be displayed for the Change Layout menu. Use the `text` attribute to provide a descriptive label for the `Layout Customizable` component so that the component is clearly visible on the page at runtime. For example: `text="Change Layout"`

Table B–3 (Cont.) Attributes of the Layout Customizable Component

Attribute	Type	Supports EL?	Description
accessKey	Char	Yes	A character used to gain quick access to this button. For accessibility reasons, this functionality is not supported in screen reader mode. If the same access key appears in multiple input fields in the same page of output, the rendering user agent cycles among the elements accessed by the similar keys. Note that user agents are inconsistent about dealing with two links having same access key, and so the cycling behavior is dependent on what the user agent provides. This attribute is sometimes referred to as the "mnemonic". The character specified by this attribute must exist in the `Text` attribute of this button instance. If it does not, the user receives no visual indication of the existence of the `accessKey`. The easiest, and most convenient way to specify both the text and the mnemonic together is to use `textAndAccessKey`. Note that the `accessKey` is triggered by browser-specific and platform-specific modifier keys. It even has browser-specific meaning. For example, Internet Explorer 7.0 sets focus when you press Alt+<accessKey>. Firefox 2.0 on some operating systems sets focus when you press Alt+Shift+<accessKey>. Firefox 2.0 on other operating systems set focus when you press Control+<accessKey>. Refer to your browser's documentation for how it treats accessKey.
showIcon	Boolean	Yes	Specifies whether the default **Change Layout** icon is visible or not. The default value is `true`.
showLayoutChanger	Boolean	Yes	Specifies whether the **Change Layout** icon or text must be displayed. The default value is `true`. When set to `true`, the runtime behavior of the layout changer is as follows: ■ Displayed in View mode. ■ Displayed in Edit mode. It is always displayed in Edit mode, even if the value of the `showLayoutChanger` attribute is `false`. ■ If there is a security or MDS restriction on the Layout Customizable component, then the layout changer is not displayed in View or Edit modes.

Table B–3 (Cont.) Attributes of the Layout Customizable Component

Attribute	Type	Supports EL?	Description
showTypes	String	Yes	Specifies the names of predefined layouts that must be displayed in the layout changer. Use this attribute to control the layout options displayed to users. You can provide a space-separated list of layout types. By default, all eight layouts are displayed when a user clicks the Change Layout icon or link.
shortDesc	String	Yes	The short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the `shortDesc` is displayed in a note window.
Style Attributes			
styleClass	String	Yes	A CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle`'s CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
Behavior Attributes			
binding	`oracle.adf.view.page.editor.component.LayoutCustomizable`	Supports only EL	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean. For example: `binding="#{yourManagedBean.Binding}"`

Supported Facets

Table B–4 describes the facets provided for a `Layout Customizable` component.

Table B–4 Facets of the Layout Customizable Component

Name	Description
contentA	Prepopulated with a `Panel Customizable` component. This `Panel Customizable` contains content to be rendered in area A in the layouts shown in Table B–5.
contentB	Prepopulated with a `Panel Customizable` component. This `Panel Customizable` contains content to be rendered in area B in the layouts shown in Table B–5.
facetSeparator	Content to be rendered once between each facet.
separator	Content to be rendered once between each of the other children.

Predefined Layout Types

When you add a `Layout Customizable` component to the page, three child `Panel Cutsomizable` components are added by default; one direct child and two `Panel Customizables` in the two facets. However, you can replace the child `Panel Customizable` components with any container components of your choice. When you select the layout type, the components in these `Panel Customizables` are arranged according to the layout type chosen.

Table B–5 describes the eight layouts that you can apply to your page or an area of the page at design time. To easily describe how components are laid out, let us assume the following:

- The child `Panel Customizable` of `contentA` facet is called **A**.
- The child `Panel Customizable` of `contentB` facet is called **B**.
- The primary `Panel Customizable`, which is a direct child of the `Layout Customizable`, is called **C**.

Table B–5 Values for Type Attribute of Layout Customizable

Layout Type	Image Showing Content Arrangement
oneColumn	 **Note**: In a `oneColumn` layout, `A` and `B` are rendered only if they contain child components.
threeColumn	
threeColumnNarrow	
twoColumn	
twoColumnBottom	
twoColumnNarrowLeft	
twoColumnNarrowRight	
twoColumnTop	

B.1.4 Panel Customizable Component

Use the `Panel Customizable` component as a container for page content that can be customized at runtime. You can add content inside a `Panel Customizable` component at runtime using the Resource Catalog. When you add child `Show Detail Frame` components inside a `Panel Customizable`, the `Show Detail`

`Frame` components provide runtime customization options for arranging or deleting components on the page.

Use this component only to perform runtime customizations on child components. If you just want a container to arrange components, then it is recommended that you use an ADF Faces container like `Panel Group Layout`.

Geometry Management

The `Panel Customizable` component can be stretched by a parent layout component that stretches its children, for example, a `Panel Stretch Layout` or `Panel Splitter` component. By setting the `layout` attribute on the `Panel Customizable` component to `stretch`, the child component can be stretched to the size of the `Panel Customizable`.

Table B–6 describes the attributes of a `Panel Customizable` component.

Table B–6 Attributes of a Panel Customizable Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ▪ Must not be a zero-length String. ▪ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ▪ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.
halign	String	Yes	Specifies the horizontal alignment of child components. You can set horizontal alignment on child components only if the `Panel Customizable` component has a horizontal or vertical layout. Available values are `center`, `end`, `left`, `right`, and `start`. Default value is `start`.

Table B–6 (Cont.) Attributes of a Panel Customizable Component

Attribute	Type	Supports EL?	Description
valign	String	Yes	Specifies the vertical alignment of child components. You can set vertical alignment on child components only if the `Panel Customizable` component has a horizontal layout. Available values are `baseline`, `bottom`, `middle`, and `top`. Default value is `top`.
shortDesc	String	Yes	Specifies a short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent. For example, Firefox 2 truncates long tooltips.

Table B–6 (Cont.) Attributes of a Panel Customizable Component

Attribute	Type	Supports EL?	Description
layout	String	Yes	Specifies how the children of the `Panel Customizable` must be laid out. Available options are `horizontal`, `vertical`, `scroll`, `stretch`, and `auto`. The default value is `vertical`. Depending on the value of the `layout` attribute, child components are laid out as follows: ■ If you select `vertical`, then the child components are displayed one below the other and can be moved either up or down within the layout. ■ If you select `horizontal`, then the child components are displayed adjacent to each other and can be moved either to the left or right within the layout. ■ If you select `scroll`, then the child components are displayed one below the other, with scroll bars displayed if content overflows. ■ If you select `stretch`, then the first child component is stretched to fill up available space in the `Panel Customizable` component. Subsequent child components are ignored. However, they are not removed from the page. The Add icons for adding new `Panel Customizable` components adjacent to the selected one are not enabled if you set `layout` to `stretch`. ■ If you select `auto`, the child component is stretched only if the `Panel Customizable` is stretched by its parent. If not, the content scrolls.
Style Attributes			
styleClass	String	Yes	A CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.

Table B–6 (Cont.) Attributes of a Panel Customizable Component

Attribute	Type	Supports EL?	Description
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle` attribute's CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
Behavior Attributes			
allowAction	Boolean	Yes	Determines whether this component allows addition of child components, dragging and dropping `Show Detail Frame` components inside and from it, and rearranging child components at runtime. It also determines whether the Add Box icons (that enable users to add Box components adjacent to the component) must be rendered on the component. Available values are `none` and `all`. The default value is `all`. **Note:** You can also set this attribute while defining component security in the application's `adf-config.xml` file. For more information, see Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."
showEditAction	Boolean	Yes	While in Edit mode, in Design view, renders an **Edit** icon on the `Panel Customizable` header that enables users to edit component properties at runtime. Available values are `true` and `false`. Default value is `true`. This is relevant only in a Composer-enabled page, that is, if the `Panel Customizable` component is nested within a `Page Customizable` component.

Table B–6 (Cont.) Attributes of a Panel Customizable Component

Attribute	Type	Supports EL?	Description
showSplitAction	Boolean	Yes	In Design view of a page in Edit mode, renders the Add Box icons on the `Panel Customizable` header that enable users to add Box components above, below, to the left, or right of the selected component. Available values are `true` and `false`. Default value is `true`. This is relevant only in a Composer-enabled page, that is, if the `Panel Customizable` component is nested within a `Page Customizable` component. However, the Add Box icons are not displayed on the root `Panel Customizable` component. Although the default value for this attribute is `true`, the Add Box icons are not displayed on a component that is stretched.
showTabAction	Boolean	Yes	In Design view of a page in Edit mode, renders the Add a Tab Set or a Tab icon on the `Panel Customizable` header that enables users to add a tabs to the page. Available values are `true` and `false`. Default value is `false`. This is relevant only in a Composer-enabled page, that is, if the `Panel Customizable` component is nested within a `Page Customizable` component.
Advanced Attributes			
binding	`oracle.adf.view.page.editor.component.PanelCustomizable`	Supports only EL	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean.

Supported Facets

The `Panel Customizable` component supports a `separator` facet, which can be used to render content once between each of its child components.

Disabling the Edit Option on a Panel Customizable Component

When you switch to the Edit mode of a page at runtime, in Design view you will notice an **Edit** icon on all `Panel Customizable` or `Box` components that can be edited. To disable property editing on a specific `Panel Customizable` or `Box` component in Design view, you must set the `showEditAction` attribute on the component to `false` at design time. The **Edit** icon is no longer displayed on the

component in Design view. However, users can still select such a component in Source view and edit its properties.

B.1.5 Show Detail Frame Component

Add `Show Detail Frame` components inside `Panel Customizable` components to enable runtime customizations like move, minimize, restore, and delete of child components. You can move `Show Detail Frame` components only if they are added inside a `Panel Customizable` component.

Geometry Management

A `Show Detail Frame` component can be stretched by a parent layout component that stretches its children, for example, `Panel Customizable`.

A `Show Detail Frame` component stretches its child component to fill the height and width available to it. It is recommended that you add only one child component inside a `Show Detail Frame` component. However, if you have more than one child component, the first child component is stretched to the height and width of the `Show Detail Frame`'s content area and subsequent child components are ignored.

The resize handler at the lower right corner of a `Show Detail Frame` enables you to resize the component vertically. From then on, the `Show Detail Frame` cannot be stretched by its parent.

Table B–7 describes the attributes of a `Show Detail Frame` component.

Table B–7 Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ▪ Must not be a zero-length String. ▪ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ▪ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
text	String	Yes	A title for the `Show Detail Frame` component. You can change the style of this text using the `headerStyle` attribute.

icon	String	Yes	If you decide to add an icon on the header of the `Show Detail Frame` component, then this specifies the URI for the image to be used.
			For example:
			`icon="http://source-pc/images/accessability.gif"`
			Note: An image that is stored at the document root does not require a full path. For example:
			`icon="detail.gif"`

Appearance Attributes			
shortDesc	String	Yes	The short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the `shortDesc` is displayed in a note window.
background	String	Yes	Working with the skin CSS, provides a means of applying a different look and feel for this `Show Detail Frame` instance. Available values are `light`, `medium`, `dark`, and `coreDefault`. The default value is `medium`. The `coreDefault` value is similar to `medium`, but additionally it provides a rounded border for the container and renders a shadow. The background in this case would be similar to that of an ADF Faces `panelBox` component.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
displayHeader	Boolean	Yes	Indicates whether the header of the `Show Detail Frame` is displayed. The default value is `true`. If you choose to set `displayHeader` to `false` and if you have exposed some actions on the component, then a toolbar is displayed when you move the mouse over the component area. The toolbar contains a drop down icon, which displays a menu of available options. This toolbar displays only in Edit mode, if there are actions available on the component.
displayShadow	Boolean	Yes	Specifies whether a shadow is cast by the `Show Detail Frame` component. The default value is `false`.
expansionMode	String	Yes	The default state of the `Show Detail Frame`. Available values are `minimized` and `normal`. The default display mode is `normal`.
Actions Attributes			
displayActions	String	Yes	Specifies when seeded interactions are displayed. Available options are `onHover` and `always`. The default is `always`.
inheritGlobalActions	Boolean	Yes	This attribute is of significance if you add child custom action components, and the attribute specifies whether the global `action` value of a `custom action` must override the local value. The default value is `false`. For information about global custom actions, see Section 22.3.2.2, "Defining Custom Actions at the Global Level."
showMoveAction	String	Yes	Specifies whether the Move action is displayed in the Actions menu. Available values are `menu` and `none`. The default value is `menu`.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
showRemoveAction	String	Yes	Renders a **Remove** icon on the `Show Detail Frame` header. Clicking this icon removes the component from the page. Available values are `chrome` and `none`. The default value is `none`. Note: In page Edit mode, the Remove icon appears regardless of this setting, provided other restrictions have not been applied.
showResizer	String	Yes	Specifies whether a resize handle is displayed on the lower right corner of the `Show Detail Frame`. You can alter only the height of a `Show Detail Frame` while resizing it. Available values are `never` and `always`. The default value is `always`.
showMinimizeAction	String	Yes	Specifies whether the minimize action is displayed on the header. Available values are `chrome` and `none`. The default is `chrome`.
showEditAction	Boolean	Yes	While in Edit mode, in Design view, renders an **Edit** icon on the `Show Detail Frame` header that enables you to edit component properties at runtime. Default value is `true`. This is relevant only in a Composer-enabled page, that is, if the `Show Detail Frame` component is nested within a `Page Customizable` component.
Style Attributes			
contentStyle	String	Yes	The CSS style to apply to the `Show Detail Frame` content area. Manually enter any style in compliance with CSS version 2.0 or later. Th CSS styles in this attribute are commonly used to define the height and background color of the `Show Detail Frame`. For example, `height:200px; background-color:green;` **Note**: If the `Show Detail Frame` component is nested inside a `Panel Customizable` component with a horizontal layout, you must specify a width for the `Show Detail Frame` component to ensure that it is rendered in page View mode.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
styleClass	String	Yes	The CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle`'s CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
headerStyle	String	Yes	The CSS style to apply to the `Show Detail Frame` title that is displayed on the header. Manually enter any style in compliance with CSS version 2.0 or later. Th CSS styles in this attribute are commonly used to define the font, size, and color of the title text.
Behavior Attributes			
partialTriggers	String	Yes	The IDs of the components that should trigger a partial update. This component listens on the trigger components. If a trigger component receives an event that causes it to update in some way, this component requests to be updated too. Identifiers are relative to the source component (this component), and must account for NamingContainers. If your component is inside of a naming container, you can use a single colon to start the search from the root of the page, or multiple colons to move up through the NamingContainers - "`::`" pop out of the component's naming container (or itself if the component is a naming container) and begin the search from there, "`:::`" pop out of two naming containers (including itself if the component is a naming container) and begin the search from there.
disclosureListener	`javax.el.MethodExpression`	Yes	A method reference to a disclosure listener. A disclosure event is fired when the user expands or collapses the `Show Detail Frame` component.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
stretchContent	String	Yes	Specifies whether the `Show Detail Frame` stretches its child component. Available options are `true`, `false`, and `auto`. The default value is `true`. You can use the `stretchContent` attribute to stretch a child component that supports being stretched.For example, container components like `Panel Form Layout`, `Panel Customizable`, and `Show Detail Item` provide the capability to be stretched. If you select `true`, then the child component is stretched to the height and width of the `Show Detail Frame`'s content area. The default height of the `Show Detail Frame` is 200px. The height of the `Show Detail Frame` is defined using the `ContentStyle` attribute. If you select `auto`, the child component is stretched only if the `Show Detail Frame` is wrapped inside a component that stretches it. If not, `stretchContent` is set to `false`.
Advanced Attributes			
binding	`oracle.adf.view.page.editor.component.ShowDetailFrame`	Supports only EL	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean.
attributeChangeListener	`javax.el.MethodExpression`	Yes	A method reference to an attribute change listener. Attribute change events are not delivered for any programmatic change to a property. They are only delivered when a renderer changes a property without the application's specific request. An example of an attribute change event might include the width of a column that supported client-side resizing.

Table B–7 (Cont.) Attributes of a Show Detail Frame Component

Attribute	Type	Supports EL?	Description
selectChild	String	Yes	Specifies whether runtime selection and customization are enabled on child components. This is relevant only in a Composer-enabled page, that is, if the `Show Detail Frame` component is nested within a `Page Customizable` component. Default value is `yes`. However, when you add a task flow as a child of the `Show Detail Frame` component, this attribute defaults to `no`. If you select `yes`, then child components can be selected and customized at runtime.
Other Attributes			
helpTopicID	String	Yes	Specifies a Help topic ID that is used to link to a custom Help topic from the component. To link to a Help topic from your `Show Detail Frame` component instance, you can either link to an HTML file that is stored directly in your application directory, or to a Help topic that is part of a JAR file.

Supported Facets

Table B–8 describes the facets available on a `Show Detail Frame` component.

Table B–8 Show Detail Frame Facets

Name	Description
titleBarAction	Used if an action is to be associated with title of the `Show Detail Frame` component. The component specified by the facet is rendered in place of the `Show Detail Frame`'s `text` attribute.
additionalActions	Used if some additional actions are to be added to the Actions menu available on the `Show Detail Frame` header. To ensure that the additional actions appear similar to other actions on the menu, use `Command Menu Item` components or a `Group` component with nested `Command Menu Item` components inside this facet.

Displaying Child Component Properties Along with Show Detail Frame Properties

When you switch to the Edit mode of a page at runtime, in Design view you will notice an **Edit** icon on all `Show Detail Frame` components that can be edited. Clicking this icon invokes the Component Properties dialog in which users can edit the `Show Detail Frame`'s properties. If you want the Component Properties dialog to display the `Show Detail Frame`'s properties and its child component's properties, then you can use the custom attribute, `sdf_selection_rule`.

In JDeveloper, right-click the `Show Detail Frame` component and select **Insert inside cust:showDetailFrame**, **JSF core**, and then **Attribute**. In the Insert Attribute

dialog, specify `sdf_selection_rule` as the name and `sdf_for_edit_mode_only` as the value. In Source mode, this attribute appears as shown in the following example:

```
<cust:showDetailFrame text="showDetailFrame 1" id="sdf1">
  <f:attribute name="sdf_selection_rule" value="sdf_for_edit_mode_only"/>
</cust:showDetailFrame>
```

Disabling the Edit Option on a Show Detail Frame Component

When you switch to the Edit mode of a page at runtime, in Design view you will notice an **Edit** icon on all `Show Detail Frame` components that can be edited. To disable property editing on a specific `Show Detail Frame` or `Movable Box` component in Design view, then you can do this by setting the `showEditAction` attribute on the component to `false` at design time. The **Edit** icon is no longer displayed on the component when you enter the Edit mode of the page at runtime. However, users can still select such a component in Source view and edit its properties.

B.1.6 Custom Action

Use `Custom Action` components to trigger navigational flow in a task flow when a region is included inside a `Show Detail Frame` component. You can define custom actions corresponding to the ADFc outcomes of the task flows. At runtime, these custom actions are displayed on the `Show Detail Frame` header as icons, menu options, or both. A `Custom Action` must always be defined as a child component of a `Show Detail Frame` and is useful only when the `Show Detail Frame` also contains a task flow as its child component.

You can add as many `Custom Action` components as there are ADFc outcomes in the task flow. If you add a `Custom Action` attribute without a corresponding ADFc outcome, or if the `action` attribute value does not match an ADFc outcome defined on the view that is currently being displayed by the task flow, then that action is not displayed at runtime.

Note: You can also define custom actions for `Show Detail Frame` components at the global level. For more information, see Section 22.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow."

Table B–9 describes the attributes of a `Custom Action` component.

Table B–9 Attributes of a Custom Action Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ■ Must not be a zero-length String. ■ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ■ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
action	String	Yes	Specifies the action outcome defined for the task flow. The value of this attribute must match the relevant ADFc outcome in the task flow definition file.
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.
text	String	Yes	Represents the text of the menu item link. This is applicable for a custom action rendered as a menu item link.
Appearance Attributes			

Table B–9 (Cont.) Attributes of a Custom Action Component

Attribute	Type	Supports EL?	Description
icon	String	Yes	For a custom action rendered as a toolbar link, specifies the icon to be rendered on the chrome. For a custom action rendered as a menu item link, this represents the icon to be displayed against the menu item text. This attribute supports these various types of URIs: ■ absolute - an absolute path to the image, like `"http://oracleimg.com/admin/images/ocom/hp/oralogo_small.gif"` ■ relative - a path located relatively to the source page, like `"bullet.jpg"` ■ context relative - a path relatively based on the web application's context root, like `"/images/error.png"` ■ server relative - a path relatively based on the web server by application name, like `"//adf-richclient-demo-context-root/images/error.png"` As icon does not allow alternative text to be provided for the image, in order to create an accessible product an icon must only be used when its use is purely decorative.
shortDesc	String	Yes	Provides a short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the `shortDesc` is displayed in a note window. This is applicable only for a custom action rendered as a toolbar icon.
location	String	Yes	Determines whether the custom action is rendered as a toolbar icon or a menu item link. Available values are `chrome`, `menu`, or `both`. Default value is `chrome`.
Other Attributes			
actionComponent	String	Yes	Specifies the ID of the command component that must be queued for the action event. When the `actionComponent` attribute is specified, the `Show Detail Frame` component queues the action event on this component. For more information, see Section 22.3.2.3, "Configuring Custom Actions that Display Task Flow Views in a Separate Browser Window."

B.1.7 Show Property

Use the `Show Property` component to display an attribute or parameter in a custom property panel at runtime. Use one `Show Property` component for each attribute or parameter that you want to display to users at runtime.

For more information about the `Show Property` component, see Section 21.3.6, "How to Display Properties and Parameters in a Custom Property Panel."

Table B–10 describes the new attributes of a `Show Property` component.

Table B–10 Attributes of a Show Property Component

Attribute	Type	Supports EL?	Description
Common Attributes:			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ▪ Must not be a zero-length String. ▪ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ▪ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`. The rendering of a component can also be defined at runtime using the Show Component and Hide Component options.
attributeType	String	Yes	Specifies the property type. Supported values are `text`, `cbox`, `date`, `lov`.
value	String	Yes	Specifies the current value of the property. If the EL binding for the "value" points to a bean property with a getter but no setter, and this is an editable component, the component will be rendered in read-only mode.
label	String	Yes	Specifies the label of the component. If you want the label to appear above the control, use a `panelFormLayout`.
Data Attributes:			
selectItem			Specifies the list of values to be displayed if `attributeType` was set to `lov`. This attribute takes an EL value that returns the list of values dynamically.
Appearance Attributes:			

Table B–10 (Cont.) Attributes of a Show Property Component

Attribute	Type	Supports EL?	Description
columns	Int	Yes	Specifies the size of the text control specified by the number of characters shown. The number of columns is estimated based on the default font size of the browser.
rows	Int	Yes	Specifies the height of the text control specified by the number of characters shown. The default value is 1, which generates a one-row input field. The number of rows is estimated based on the default font size of the browser.
shortDesc	String	Yes	Provides a short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the `shortDesc` is displayed in a note window. This is applicable only for a custom action rendered as a toolbar icon.
simple	Boolean	Yes	A boolean value that controls whether the component provides label support; when set to "true", the component will not display the label (these may be ignored: label, labelAndAccessKey, accessKey, showRequired, help facet) and may use simpler layout primitives. One of the usecases can be when the component is used in repeatable elements like in table, for-each etc., where label is not required.
unselectedLabel			Specifies the label for the option that represents a value of null, meaning nothing is selected. If `unselectedLabel` is not set and if the component does not have a selected value, then an option with an empty string as the label and value is rendered as the first option in the choice box (if there isn't an empty option already defined). Once an option has been successfully selected, and if `unselectedLabel` is not set, then the empty option will not be rendered.
Behavior Attributes:			
required	Boolean	Yes	Specifies whether a non-null, non-empty value must be entered. If `false`, validators will not be executed when the value is null or empty.
autoSubmit	Boolean	Yes	Specifies an attribute which if set to TRUE on a form element, the component will automatically submit when an appropriate action takes place (a click, text change, etc.). Since `autoSubmit` is a partial submit, also submitted and re-rendered are any other components with `partialTriggers` pointing to this component.

Table B–10 (Cont.) Attributes of a Show Property Component

Attribute	Type	Supports EL?	Description
disabled	Boolean	Yes	Specifies whether the property should be shown as disabled. This is useful if the `Show Property` component is inside an iterator whose model specifies that the property must be shown as disabled, for example, if an MDS restriction has been applied on the property.
valueChangeListener	javax.faces.el.MethodBinding	EL only	Specifies a method reference to a value change listener.
Advanced Attributes:			
binding	`oracle.adf.view.rich.component.fragment.UIXDeclarativeComponent`	Supports only EL	Specifies an EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean.
clientComponent	Boolean	Yes	Specifies whether a client-side component will be generated. A component may be generated whether or not this flag is set, but if client Javascript requires the component object, this must be set to true to guarantee the component's presence. Client component objects that are generated today by default may not be present in the future; setting this flag is the only way to guarantee a component's presence, and clients cannot rely on implicit behavior. However, there is a performance cost to setting this flag, so clients should avoid turning on client components unless absolutely necessary.
componentId			If you use the `Show Property` tag in Composer's property panel, `componentId` specifies the ID of the component for which the property must be rendered.
showELBuilder	Boolean	Yes	Specifies whether to display the EL builder for the property.
showResourceEditor	Boolean	Yes	Specifies whether to display the resource string editor for the property. However, this option is enabled only if you have configured resource string editing in the application and the property is of type String. For more information, see Section 22.12, "Configuring Runtime Resource String Editing."
taskFlowId	String	Yes	If `Show Property` is used to display task flow parameters, `taskFlowId` specifies the ID of the task flow.
taskFlowParam	String	Yes	If `Show Property` is used to display task flow parameters, `taskflowParam` specifies the task flow parameter that the input field must display.
validator	String	Yes	Specifies a method reference to a validator method.

Supported Facets

The `Show Property` component supports a `customLink` facet, which can be used to include a `Commamd Menu Item`. The `Command Menu Item` specified in this facet appears as a custom link in the dropdown next to the property field, along with Resource String editor and EL Builder.

B.2 Composer-Specific Files and Configurations

This section describes the various files that you can modify to extend Composer capabilities. While some files, for example the Composer extension file, `pe_ext.xml`, are entirely relevant to Composer only, others are ADF configuration files that can take entries for Composer-specific configurations.

This section includes the following topics:

- Section B.2.1, "pe_ext.xml"
- Section B.2.2, "adf-config.xml"
- Section B.2.3, "adf-settings.xml"
- Section B.2.4, "web.xml"

B.2.1 pe_ext.xml

You can create a `pe_ext.xml` file and add elements in it to register new Composer add-ons and custom property panels, selectively render panels, register event handlers, and define property filters. This file is not available by default when you add Oracle Components to a page. You can create this file in the `META-INF` directory in your `Portal` project; for example, in the *`APPLICATION_HOME`*`\Portal\src\META-INF` directory. This section describes the different elements of the Composer extension file schema and explain when you may want to use the schema elements.

> **Note:** In applications that use shared libraries with their own pe_ext.xml file, use the **`sequence`** element within the `pe-extension` tag to define the order of execution. Use `sequence="first"` to lay down default settings, and `sequence="last"` to load the file last and override settings defined by files loaded previously.

This section contains the following subsections:

- Section B.2.1.1, "addon-config"
- Section B.2.1.2, "property-panels"
- Section B.2.1.3, "lov-config"
- Section B.2.1.4, "event-handlers"
- Section B.2.1.5, "drop-handlers"
- Section B.2.1.6, "filter-config"
- Section B.2.1.7, "elbuilder-config"
- Section B.2.1.8, "selection-config"
- Section B.2.1.9, "Sample pe_ext.xml File"

B.2.1.1 addon-config

Use the `addon-config` element to include entries for new Composer add-ons and custom property panels. For example, to add a new **About** button to your page that invokes a panel displaying information about your application, you must first create a task flow containing the information and then register this task flow in the `pe_ext.xml` file. Composer add-ons declared inside the `addon-config` element are configured in the `adf-config.xml` file, whereas custom property panels declared within the `addon-config` element are configured within the `property-panels` element in the extension file itself.

For information about configuring add-ons in `adf-config.xml`, see Section B.2.2.2, "addon-panels." For information about configuring custom property panels using the `property-panels` element, see Section B.2.1.2, "property-panels."

Insert the `addon-config` element within the `pe-extension` tag. You can add only one `addon-config` element in the XML file. Within the `addon-config` element you can have a `panels` element to define add-ons.

panels

Use the `panels` element to register new add-ons or property panels in Composer. Add the `panels` element within the `addon-config` element. You can have only one `panels` element in an extension file, and all custom panels must be defined inside this element.

panel

Use `panel` elements to define individual add-ons and property panels to display to users. Within the `panels` element you can insert any number of individual `panel` elements that define the add-ons or property panels to be displayed in Composer. You must have one `panel` element for every add-on or property panel you want to display.

Table B–11 describes the attributes of the `panel` element.

Table B–11 panel Element Attributes

Attribute	Description
`name`	An identifier that is used while referencing the new add-on in `adf-config.xml`, or a property panel in the `property-panels` section. This must be unique in the application.
`title`	The title to display on the button used to invoke the task flow in Composer. You can use EL for this property to show a localized title. For custom property panels, the title to display on the new tab for that panel.
`icon`	The icon that appears next to the title on the button. This is optional.
`taskflow-id`	The ID of the task flow that defines the add-on or property panel.
`help-topic-id`	A help topic ID that links to a custom help topic from the add-on or property panel. This is optional.

When a `panel` element is deleted from the file, that add-on or property panel is not displayed on the page.

The following example shows how to use the `addon-config`, `panels`, and `panel` elements:

```
<addon-config>
  <panels>
    <panel name="oracle.fod.custom.panel" title="About FOD"
           icon="adf/webcenter/images/about.gif"
           taskflow-id="/WEB-INF/about-fod.xml#about-fod"
    />
  </panels>
</addon-config>
```

For more information about add-ons, see Section 21.2, "Creating Composer Add-Ons."

Optionally, to register an event handler for a specific add-on panel, you can also add an `event-handlers` element inside a `panel` element, as shown in the following example:

```
<event-handlers>
  <event-handler event="close">
    oracle.fod.custom.TaskFlowEventHandler
  </event-handler>
</event-handlers>
```

The XML code is the same as for any generic event handler. For more information, see Section B.2.1.4, "event-handlers."

B.2.1.2 property-panels

Use the `property-panels` element to register custom property panels. Add the `property-panels` element within the `addon-config` element. For information about custom property panels, see Section 21.3, "Creating Custom Property Panels." You can have only one `property-panels` element in an extension file.

property-panel

Use the `property-panel` element to define a custom property panel to display for a particular component in Composer. Within the `property-panels` element you can insert any number of individual `property-panel` elements to define custom property panels to display as tabs in the Component Properties dialog. To specify the component or task flow for which you want to register a custom property panel, you must add a `component` or `taskflow-id` element respectively within the `property-panel`.

Table B–12 describes the `property-panel` element attributes.

Table B–12 property-panel Element Attributes

Attribute	Description
`name`	An identifier for the custom property panel you want to display in the Component Properties dialog.
`rendered`	Whether this panel is displayed to users in the Component Properties dialog. This can take an EL value.

component

Use the `component` element to specify the fully qualified class name of the component for which you are registering the custom property panel. You can have only one `component` inside a `property-panel` element.

taskflow-id

Use the `taskflow-id` element to specify the name of the task flow for which you are registering the custom property panel. You can have only one `taskflow-id` inside a `property-panel` element.

panel

Use the `panel` element to specify a custom property panel to display as a tab in the Component Properties or Page Properties dialog for the component. Within a `property-panel` element, you can have any number of `panel` elements. Table B–13 describes the attributes of a `panel` element.

Table B–13 panel Element Attributes

Attribute	Description
`name`	Name with which you declared the panel in the `addon-config` section.
`rendered`	Whether the custom property panel must be rendered or not.
`parameter`	Parameters to be passed to the implementing task flow. Takes an EL only and must return a `Map<String, String>`.

When a `panel` element is deleted from the file, that property panel is no longer displayed in the Component Properties dialog.

The following example shows how to register a custom property panel for a `Command Button` component:

```
<property-panels>
  <property-panel name="cmdbtn">
    <component>oracle.rich.CommandButton</component>
    <panel name="prop.panel.cmdbtn"/>
  </property-panel>
</property-panels>
```

The following example shows how to register a custom property panel for a task flow:

```
<property-panels>
  <property-panel name="dashboard">
    <taskflow-id>/WEB-INF/dashboard-taskflow#prop-panel</taskflow-id>
    <panel name="dashboard.prop-panel" />
  </property-panel>
</property-panels>
```

B.2.1.3 lov-config

Use the lov-config element to register a custom List of Values or picker for a task flow parameter. Add the `lov-config` element directly under the `pe-extension` tag. You can have only one `lov-config` element in an extension file. Within this element, you can add a `task-flow-definition` element for each task flow that you want to register for LOV usage. Add an `input-parameter-definition` for each parameter that provides an LOV. Use `enumeration` elements to list the LOV options for a parameter, as shown in the following example:

```
<lov-config>
  <task-flow-definition
taskflow-id="/WEB-INF/employee-details.xml#employee-details">
    <input-parameter-definition>
      <name>Designation</name>
        <enumeration>
```

```
            <item>
              <name>Accounts Manager</name>
              <value>Accounts Manager</value>
              <description>Accounts Manager</description>
            </item>
            <item>
              <name>Accounts Assistant</name>
              <value>Accounts Assistant</value>
              <description>Accounts Assistant</description>
            </item>
            <item>
              <name>Payroll Executive</name>
              <value>Payroll Executive</value>
              <description>Payroll Executive</description>
            </item>
            <item>
              <name>Senior Manager</name>
              <value>Senior Manager</value>
              <description>Senior Manager</description>
            </item>
          </enumeration>
      </input-parameter-definition>
    </task-flow-definition>
</lov-config>
```

For more information, see Section 21.5, "Configuring Custom LOVs or Pickers for Task Flow Parameters."

B.2.1.4 event-handlers

Use the `event-handlers` element to register handlers for events generated by Composer. Add the `event-handlers` element directly under the `pe-extension` tag. You can have only one `event-handlers` element under `pe-extension`.

Alternatively, to register an event handler for a specific Composer add-on panel, you can also add an `event-handlers` element inside a `panel` element under the `addon-config` section. For more information, see Section B.2.1.1, "addon-config."

event-handler

Use the `event-handler` element to register an event handler in Composer. Within `event-handlers` you can have any number of `event-handler` elements to register handlers for save, close, deletion, addition, and selection events. For example, if you have implemented the `processSave` method in a Java class called `SaveHandler` to perform a specific action when a Save event is invoked, then you can register that implementation in Composer by adding an `event-handler` entry in the `pe_ext.xml` file as follows:

```
<event-handlers>
  <event-handler event="save">view.SaveHandler</event-handler>
</event-handlers>
```

For information about configuring event handlers, see Section 21.6, "Configuring Event Handlers for Composer UI Events."

B.2.1.5 drop-handlers

Use the `drop-handlers` element to register drop handlers to handle addition of components to the page from the Resource Catalog. Add the `drop-handlers`

element directly under the pe-extension tag. You can have only one drop-handlers element in an extension file.

drop-handler

Use the drop-handler element to register a drop handler in Composer. Within drop-handlers you can have any number of drop-handler elements to register drop handlers that you created in the application. For example, if you created a drop handler called TestDropHandler to handle addition of XML components on the page, then you can register that implementation in Composer by adding a drop-handler entry in the pe_ext.xml file as follows:

```
<drop-handlers>
  <drop-handler>test.TestDropHandler</drop-handler>
</drop-handlers>
```

For more information, see Section 21.7, "Configuring Drop Handlers in the Resource Catalog."

B.2.1.6 filter-config

Use the filter-config element to define property filters for components that can be edited at runtime. For a selected component, the Component Properties dialog displays properties that can be edited. If you do not want to expose all properties for editing, you can filter specific properties so that they are not displayed in the Component Properties dialog. You can specify global-level filters to filter common attributes across all components, or you can define tag-level filters to filter properties for a particular component only.

Add the filter-config element directly under the pe-extension tag. You can have only one filter-config element in an extension file.

For information about property filters, see Section 21.8, "Defining Property Filters."

global-attribute-filter

Use the global-attribute-filter element to filter specific properties across all components. You can have only one global-attribute-filter element in the extension file. Within this element you can include any number of attribute elements to define property filters for components from different libraries.

taglib-filter

Use the taglib-filter element to define property filters for components within a particular library. You can include multiple taglib-filter elements to filter properties for components in different libraries. Within this element you can include tag elements for every component belonging to the library.

Table B–14 describes the attribute the taglib-filter element can take.

Table B–14 tag Element Attribute

Attribute	Description
namespace	Namespace of the tag library containing the component whose properties you want to filter.

tag

Use the tag element to specify a component whose properties you want to filter. You can include multiple tag elements within a taglib-filter element. Within this element you can include attribute elements for all properties you want to filter.

Table B–15 describes the attribute the `tag` element can take.

Table B–15 tag Element Attribute

Attribute	Description
`name`	Name of the component whose properties you want to filter, for example, `commandButton`, `portlet`.

attribute

Use the `attribute` element to specify a property to filter either across all components or for the specified component only. To define global-level filters, you must include `attribute` elements inside the `global-attribute-filter`. To define property filters for a particular component, you must include `attribute` elements inside the `taglib-filter` element.

Table B–16 describes the attributes the `attribute` element can take.

Table B–16 attribute Element Attributes

Attribute	Description
`name`	Name of the property you want to filter.
`filtered`	Whether the property must be filtered or not. Takes a value of `true` or `false`. To filter the property in a specific view, use the `view` attribute.
`label`	The label for the property in the UI. To retrieve localized strings or show a different label depending on the view, use EL.
`view`	(Optional) The view in which to show/hide this property. Takes a value of `design` or `source`.

The following example shows how you can define global and component-level property filters:

```
<filter-config>
   <global-attribute-filter>
    <attribute name="readOnly" filtered="false" label="Read Only"/>
    <attribute name="required" filtered="false" label="Mandatory"/>
    <attribute name="shortDesc" filtered="false" label="#{composerBundle.PNL_TB_
SHORTDESC}"/>
  </global-attribute-filter>

  <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
    <tag name="goLink">
      <attribute name="accessKey" view="design"/>
      <attribute name="targetFrame" filtered="false" view="source"/>
      <attribute name="destination" filtered="false"
label="#{pageEditorPanelBean.layoutView ?  'Where do you want to surf?' :
'Destination'}"/>
    </tag>
    <tag name="inputText">
      <attribute name="all" view="design"/>
      <attribute name="label" label="Field Name" filtered="false" />
      <attribute name="readOnly" filtered="false" />
      <attribute name="required" filtered="false" />
      <attribute name="rendered" filtered="false" />
      <attribute name="wrap" filtered="false" />
    </tag>
  </taglib-filter>
```

```
</filter-config>
```

B.2.1.7 elbuilder-config

Use the `elbuilder-config` element to configure custom values for the expression builder. The expression builder invoked from Composer's Component Properties dialog enables users to specify EL values for component properties. Users can either select for a set of predefined values or enter an EL value in the text box. You can customize the expression builder to provide more options to users. For more information, see Section 21.4, "Extending the Expression Builder."

Add the `elbuilder-config` element directly under the `pe-extension` tag. You can have only one `elbuilder-config` element in an extension file.

selector

Use `selector` elements to define and include custom options, as shown in the following example:

```
<elbuilder-config>
  <!-- define selector -->
  <selector id="CustomELParameter">
    view.CustomELParameter
  </selector>

  <!-- include selector -->
  <selectors>
    <selector id="CustomELParameter"/>
  </selectors>
</elbuilder-config>
```

You can include many `selector` elements inside a `selectors` element.

B.2.1.8 selection-config

Use the `selection-config` element to to enable or disable direct select for specific components and define associated operations. To make a component selectable, you must define operations within this tag. For more information, see Section 21.11, "Enabling Direct Select in Design View."

Within the `<selection-config>` tag, you can configure selection globally using `<global-filter>` and by namespace and tag using `<selection-taglib-filter>`.

selection

The selection tag is used to enable or disable the associated direct select operations for a specific view. In the current release, direct select options can only be enabled/disabled for the Design view.

operation

Available operations are defined within `<operation>` tags under each specific component. The following elements are used to define operations:

- `name`: Defines the operation to be executed. (Use standard or custom operations.)
- `label` (optional): Overrides the operation name as the tab label in the popup window.
- `filtered`: Set to `false` to include the operation in the popup window.

```
<selection-config>
```

```
  <global-filter>
    <selection view="design" enabled="false"/>
  </global-filter>

  <selection-taglib-filter
   namespace="http://xmlns.oracle.com/adf/faces/customizable">
    <tag name="showDetailFrame">
      <selection view="design" enabled="true"/>
      <operation name="oracle.adf.pageeditor.pane.sdfprop"
                 label="SDF Change Property"
                 filtered="false"/>
    </tag>
  </selection-taglib-filter>
</selection-config>
```

B.2.1.9 Sample pe_ext.xml File

Example B–1 shows a sample `pe_ext.xml` file with different elements used to extend Composer capabilities.

Example B–1 Sample pe_ext.xml File

```
<pe-extension xmlns="http://xmlns.oracle.com/adf/pageeditor/extension">

<addon-config>
<!-- Composer add-on panels configuration -->
  <panels>
    <panel name="oracle.fod.custom.panel" title="About FOD"
           icon="adf/webcenter/images/about.gif"
           taskflow-id="/WEB-INF/about-fod.xml#about-fod"/>
  </panels>

<!-- Composer property panels configuration -->
  <property-panels>
    <property-panel name="cmdbtn">
      <component>oracle.rich.CommandButton</component>
      <panel name="prop.panel.cmdbtn" />
    </property-panel>
  </property-panels>
</addon-config>

<Save event handler configuration -->
  <event-handlers>
    <event-handler event="save">view.SaveHandler</event-handler>
  </event-handlers>

<!-- Test drop handler configuration -->
  <drop-handlers>
    <drop-handler>test.TestDropHandler</drop-handler>
  </drop-handlers>

<!-- Property filter configuration. Properties defined here are not displayed in
Composer. -->
  <filter-config>
    <global-attribute-filter>
      <attribute name="accessKey" />
      <attribute name="attributeChangeListener" />
      <attribute name="autoSubmit" />
      <attribute name="binding" />
    </global-attribute-filter>
```

```
    <taglib-filter namespace="http://xmlns.oracle.com/adf/faces/rich">
      <tag name="commandButton">
        <attribute name="text" />
        <attribute name="icon" />
      </tag>
  </filter-config>
</pe-extension>
```

B.2.2 adf-config.xml

The `adf-config.xml` file specifies application-level settings that are usually determined at deployment and are often changed at runtime. This file is created when you create your application and is located in the `ADF META-INF` folder under `Descriptors` in the Application Resources panel. The following example shows the minimal `adf-config.xml` file created when you create a Framework application:

```
<?xml version="1.0" encoding="windows-1252" ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties">
  <adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
    <adf-property name="adfAppUID" value="Application4-2138"/>
  </adf:adf-properties-child>
</adf-config>
```

When you add the `Page Customizable` component to a page, certain required configurations are added to the `adf-config.xml` file.

You must update the `adf-config.xml` file when you perform tasks such as registering new add-ons and custom property panels, selectively rendering add-ons, creating customization layers, enabling and disabling the Composer sandbox, creating custom Resource Catalogs, and disabling Source view. The following sections describe the various `adf-config.xml` elements used for Composer-specific configurations.

This section contains the following subsections:

- Section B.2.2.1, "page-editor-config"
- Section B.2.2.2, "addon-panels"
- Section B.2.2.3, "sandbox-namespaces"
- Section B.2.2.4, "session-options-factory"
- Section B.2.2.5, "enable-design-views"
- Section B.2.2.6, "allow-el and protect-el"
- Section B.2.2.7, "rcv-config"
- Section B.2.2.8, "customizableComponentsSecurity"
- Section B.2.2.9, "mds-config"
- Section B.2.2.10, "resource-string-editor"
- Section B.2.2.11, "enable-source-view"
- Section B.2.2.12, "enable-zoom"
- Section B.2.2.13, "persistent-change-manager"
- Section B.2.2.14, "taglib-config"
- Section B.2.2.15, "security-config"
- Section B.2.2.16, "Sample adf-config.xml File"

B.2.2.1 page-editor-config

Use the <pe:page-editor-config> element to include Composer-specific configurations such as add-on panel registration, sandbox configuration, resource string editor configuration, and so on. Add the <pe:page-editor-config> element within the adf-config tag as shown in the following example:

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
. . .
  <pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
    . . .
  </pe:page-editor-config>
</adf-config>
```

B.2.2.2 addon-panels

Use the <pe:addon-panels> element to register custom add-ons and property panels in Composer. Insert the <pe:addon-panels> element within <pe:page-editor-config>. You can have only one <pe:addon-panels> element in the file, however, within <pe:addon-panels> you can have any number of <pe:addon-panel> elements to register custom panels. For more information about Composer add-ons and property panels, see Section 21.2, "Creating Composer Add-Ons" and Section 21.3, "Creating Custom Property Panels."

addon-panel

Use a <pe:addon-panel> element to register an add-on in the adf-config.xml file. A <pe:addon-panel> element must only be included inside the <pe:addon-panels> element. You can have any number of <pe:addon-panel> elements to register custom panels. The following example shows how you can register a new add-on using these elements:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  <pe:addon-panels>
    <!-- Page Properties add-on -->
    <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-settings" />

    <!-- Page Reset add-on -->
    <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset" />

    <pe:addon-panel name="oracle.fod.custom.panel" />

  </pe:addon-panels>
  . . .
</pe:page-editor-config>
```

The <pe:addon-panel> element also supports the rendered attribute. This attribute can take an EL value and specifies whether the panel must be rendered in Composer or not.

When registering custom add-ons, you must also include entries for the default add-ons. If the default add-ons are not registered, then only your custom add-on is available in Composer.

The <pe:addon-panel> element supports the following parameters:

- **rendered**: Use to specify whether the add-on must be rendered or not. The rendered attribute can take true, false, or an EL value. Use this attribute to display an add-on conditionally based on specific criteria.

- **parameters**: Use to declaratively pass parameters to the add-on task flow. The `parameters` attribute takes an EL value as shown in the following example:

```
<pe:addon-panels>
  <pe:addon-panel
name="oracle.adf.pageeditor.addonpanels.customization-manager"
parameters="#{AppUtilBean.customizationManagerParams}"/>
    . . .
</pe:addon-panels>
```

  where `AppUtilBean` provides the logic to pass parameters to the add-on task flow.

B.2.2.3 sandbox-namespaces

Use the `<pe:sandbox-namespaces>` element to register namespaces for all metadata for which you want to enable sandbox creation at runtime. For more information about enabling sandbox creation, see Section 23.2.1, "How to Enable Composer Sandbox Creation."

Insert the `<pe:sandbox-namespaces>` element within the `<pe:page-editor-config>` element. The following example shows how to use the `sandbox-namespaces` element:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .
  <pe:sandbox-namespaces>
    <pe:namespace path="/pages"/>
    <pe:namespace path="/pageDefs"/>
  </pe:sandbox-namespaces>
</pe:page-editor-config>
```

B.2.2.4 session-options-factory

When creating customization layers in your application, use the `<pe:session-options-factory>` element to register the `ComposerSessionOptionsFactory` implementation with Composer. This class contains the logic to save customizations in different layers based on the specified criteria. For more information, see Section 23.3, "Adding Customization Layers to View and Edit Modes: Example."

The following example shows how to register a class named `AppsSessionOptionsFactoryImpl`:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .

<pe:session-options-factory>view.AppsSessionOptionsFactoryImpl</pe:session-options
-factory>
</pe:page-editor-config>
```

B.2.2.5 enable-design-views

To enable Composer direct select, you must first enable the select view. For more information on direct select functionality, see Section 21.11, "Enabling Direct Select in Design View." Add the `<pe:enable-design-views>` tag inside the `<pe:page-editor-config>` tag to define the available views. The possible values are:

- `select`: Makes the direct select view the only view.
- `add-content` (default): Makes the legacy view the only view.

- `all`: Displays a toggle button to switch between views.

By default the hot key to switch between views is Ctrl-Shift-S. To change or disable the hot key, add a `<pe:component-view-switch-key>` tag inside the `<pe:page-editor-config>` tag. For example:

```
<pe:component-view-switch-key>
  <pe:enabled>true</pe:enabled>
  <pe:key-sequence>ctrl shift l</pe:key-sequence>
</pe:component-view-switch-key>
```

B.2.2.6 allow-el and protect-el

You can configure Composer to disable EL or protect existing EL from overwrite. Use `<pe:allow-el>` to define whether or not EL can be run in application pages. The default is true. Use `<pe:protect-el>` to define whether or not EL in form elements can be edited. The default is false. For more information, see Section 21.4.2, "How to Protect Expression Language."

The following example disables EL:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .
  <pe:allow-el>false</pe:allow-el>
</pe:page-editor-config>
```

B.2.2.7 rcv-config

Use the `<rcv-config>` element to register custom Resource Catalogs with Composer and to register the `ResourceCatalogSelector` implementation when using multiple Resource Catalogs.

If you want to display a custom Resource Catalog to users in place of the default one, you must first create a catalog definition file with the required content and then register this catalog definition in the Composer extension file and `adf-config.xml` file. The custom Resource Catalog is then displayed when a user clicks an Add Content button on the page. For more information, see Section 16.2, "Creating a Custom Resource Catalog."

Insert the `<rcv-config>` element within the `adf-config` tag. The following example shows how you can use the `<rcv-config>` element to register a custom catalog called `users-catalog`:

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
  . . .
  <rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
    <default-catalog catalog-name="users-catalog"/>
  </rcv-config>
</adf-config>
```

You can also configure multiple Resource Catalogs with Composer so that different catalogs are displayed to different users depending on the specified criteria. When configuring multiple Resource Catalogs, use the `<rcv-config>` element with a `<catalog-selector>` element to register the `ResourceCatalogSelector` implementation, which contains the logic for selecting a Resource Catalog.

The following example shows how you can use the `<rcv-config>` element to register a `ResourceCatalogSelector` implementation called `CatalogSelector`:

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
```

```
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
  . . .
  <rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
    <catalog-selector class-name="webcenter.CatalogSelector"/>
    <default-catalog catalog-name="default-catalog"/>
  </rcv-config>
</adf-config>
```

B.2.2.8 customizableComponentsSecurity

Use the `<customizableComponentsSecurity>` element to apply restrictions on `Show Detail Frame` actions. Insert the `<customizableComponentsSecurity>` element within the `adf-config` tag.

Within the `<customizableComponentsSecurity>` element, you can specify the `<enableSecurity>`, `<actionsCategory>`, `<actions>`, and `<custom-actions>` elements.

enableSecurity

Use the `<enableSecurity>` element to override the security inheritance behavior on `Show Detail Frame` actions. This element can take a value of `true` or `false`. If set to `true`, then the ability for a user to modify a component is first determined from the page permissions and then adjusted according to the current set of actions defined for that type of permission. If set to `false`, then all actions are available to users. A user's page permissions and actions configured in `adf-config.xml` are ignored.

actionsCategory

Use the `<actionsCategory>` element to apply restrictions on a group of `Show Detail Frame` actions simultaneously. Within the `<actionsCategory>` element, you can have `<actionCategory>` elements for the supported categories. For more information about action categories, see Section 24.5.2, "Defining Security at the Actions Category Level."

The following example shows how `actionsCategory` can be used:

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
  . . .
  <cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
    <cust:enableSecurity value="true"/>

    <cust:actionsCategory>
      <cust:actionCategory name="personalizeActionsCategory" value="false"/>
      <cust:actionCategory name="editActionsCategory" value="true"/>
    </cust:actionsCategory>

    <cust:actions>
      . . .
    </cust:actions>

  </cust:customizableComponentsSecurity>
</adf-config>
```

actions

Use the <actions> element to apply restrictions on individual actions. Within each actions element, you can have <action> elements for all supported actions.

The following example shows how <actions> can be used:

```
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
  . . .
  <cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
    <cust:enableSecurity value="true"/>

    <cust:actionsCategory>
      . . .
    </cust:actionsCategory>

    <cust:actions>
      <cust:action name="showMinimizeAction" value="true"/>
      <cust:action name="showMoveAction" value="false"/>
    </cust:actions>

  </cust:customizableComponentsSecurity>
<adf-config>
```

For more information about applying action-level restrictions, see Section 24.5, "Applying Action-Level Restrictions on Panel Customizable and Show Detail Component Actions."

custom-actions

Use the <custom-actions> element to define custom actions to display along with the other Show Detail Frame actions. You must add only one <custom-actions> element inside the customizableComponentsSecurity section. Within this you can add any number of <custom-action> elements to define individual custom actions.

> **Note:** Alternatively, you can add a <custom-actions> element inside an <adf-config-child> element instead of inside the <customizableComponentsSecurity> section.

For more information about custom actions, see Section 22.3.2, "How to Enable Custom Actions on a Show Detail Frame Enclosing a Task Flow."

B.2.2.9 mds-config

When defining type-level application customization restrictions on components, use the <mds-config> element to register a standalone XML file describing the restrictions to be applied on specific components. For information about applying type-level restrictions, see Section 24.1.1, "How to Define Type-Level Customization Policies."

Insert the <mds-config> element within the <adf-config> tag. You can add the <mds-config> element as follows:

```
<mds-config xmlns="http://xmlns.oracle.com/mds/config">
  <type-config>
    <standalone-definitions>
```

```
      <file>Directory_Name/standalone.xml</file>
    </standalone-definitions>
  </type-config>
</mds-config>
```

B.2.2.10 resource-string-editor

Use a <pe:resource-string-editor> element to enable resource string editing in your application. Insert the <pe:resource-string-editor> element within the <pe:page-editor-config> section, as shown in the following example:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .
  <pe:resource-string-editor>
    <pe:enabled>true</pe:enabled>
  </pe:resource-string-editor>
</pe:page-editor-config>
```

For more information, see Section 22.12, "Configuring Runtime Resource String Editing."

B.2.2.11 enable-source-view

Use the <pe:enable-source-view> element to enable or disable Source view in Composer. Insert the <pe:enable-source-view> element within the <pe:page-editor-config> section. Source view is enabled by default in Composer. You can disable it by setting <pe:enable-source-view> to false, as shown in the following example:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .
  <pe:enable-source-view>false</pe:enable-source-view>
</pe:page-editor-config>
```

For more information, see Section 22.8, "Disabling the Source View Menu for an Application."

B.2.2.12 enable-zoom

Use the <pe:enable-zoom> element to enable or disable the capability to zoom into task flows in Composer. Insert the <pe:enable-zoom> element within a <pe:source-view> tag in the <pe:page-editor-config> section. Task flow zoom is enabled by default in Composer. You can disable it by setting <pe:enable-zoom> to false, as shown in the following example:

```
<pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
  . . .
  <pe:source-view>
    <pe:enable-zoom>false</pe:enable-zoom>
  </pe:source-view>
</pe:page-editor-config>
```

For more information, see Section 22.9, "Disabling Task Flow Zoom Capability."

B.2.2.13 persistent-change-manager

Use this to configure the change manager to be used for persisting changes made in Composer. Insert the <persistent-change-manager> element within the <adf-faces-config> section. You must set <persistent-change-manager> to

MDSDocumentChangeManager as follows if you want to persist application customizations to MDS:

```
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  <persistent-change-manager>

<persistent-change-manager-class>oracle.adf.view.rich.change.MDSDocumentChangeMana
ger</persistent-change-manager-class>
  </persistent-change-manager>
. . .
</adf-faces-config>
```

For more information about change persistence in Composer, see Section 22.11, "Configuring the Persistence Change Manager."

For more information about the <persistent-change-manager> element, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

B.2.2.14 taglib-config

Use the <taglib-config> element to specify the component tags that must be persisted by default. This element is relevant only when you define change persistence using <persistent-change-manager> in your adf-config.xml file.

When you add Composer components to the page, the adf-config.xml file is populated with the <taglib-config> element containing a list of tags and attributes that must be persisted. You can add more elements and attributes to the default list.

For more information about this element, see *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Insert the <taglib-config> element within the <adf-faces-config> section. The <taglib-config> element appears as shown in the following example:

```
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">
  . . .
  <taglib-config>
    <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
      <tag name="showDetailFrame">
        <persist-operations>all</persist-operations>
        <attribute name="expansionMode">
          <persist-changes>true</persist-changes>
        </attribute>
        <attribute name="contentStyle">
          <persist-changes>true</persist-changes>
        </attribute>
      </tag>
      <tag name="panelCustomizable">
        <persist-operations>all</persist-operations>
      </tag>
    </taglib>
  </taglib-config>
</adf-faces-config>
```

For more information, see Section 22.11, "Configuring the Persistence Change Manager."

B.2.2.15 security-config

Use this to override Composer-specific security configurations, specifically if you want to override the default security policies with application-level checks. Insert the

<pe:security-config> element inside the <pe:page-editor-config> section. Within the security-config element, you can define <pe:security-policies>, <pe:security-policy>, <pe:policy-class>, and <pe:task-flow-security> elements.

security-policies

Use this to encompass all the security policies and policy overrides for Composer. You can have only one <pe:security-policies> element under the <pe:page-editor-config> section.

security-policy

Use this to override an existing policy. Add one <pe:security-policy> element for each policy that you want to override.

The <pe:security-policy> element supports the name and override attributes, where name is used to specify a name with which to register the custom policy and override is used to specify the name of the default policy that you want to override.

policy-class

Use this to specify the name of the custom security policy.

task-flow-security

Use this element, along with the check-permission attribute, to enable Composer to check for a task flow's permissions so that only users with appropriate privileges are allowed to edit or customize the task flow at runtime.

For more information, see Section 24.6, "Implementing Task Flow Security."

B.2.2.16 Sample adf-config.xml File

Example B–2 shows a sample adf-config.xml file with Composer-specific configurations.

Example B–2 Sample adf-config.xml File with Composer-Specific Configurations

```
<?xml version="1.0" encoding="windows-1252" ?>
<adf-config xmlns="http://xmlns.oracle.com/adf/config"
            xmlns:adf="http://xmlns.oracle.com/adf/config/properties"
            xmlns:mdsC="http://xmlns.oracle.com/adf/mds/config">
  <adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
    <adf-property name="adfAppUID" value="configfilesapp-7198"/>
  </adf:adf-properties-child>

  <mdsC:adf-mds-config version="11.1.1.000">
    <mds-config xmlns="http://xmlns.oracle.com/mds/config">
      <persistence-config>
        <metadata-namespaces>
          <namespace path="/oracle/adf/rc/metadata"
                     metadata-store-usage="WebCenterFileMetadataStore"/>
          <namespace path="/persdef/"
                     metadata-store-usage="WebCenterFileMetadataStore"/>

          <!-- Namespace definitions for Composer sandbox -->
          <!-- Your jspx customizations alone go here -->
          <namespace path="/pages"
metadata-store-usage="WebCenterFileMetadataStore">
            <namespace-restriction type="CUSTOMIZATIONS"/>
          </namespace>
```

```
          <!-- Your pagedef customizations alone go here -->
          <namespace path="/pageDefs"
metadata-store-usage="WebCenterFileMetadataStore">
            <namespace-restriction type="CUSTOMIZATIONS"/>
          </namespace>
        </metadata-namespaces>
        <metadata-store-usages>
          <metadata-store-usage id="WebCenterFileMetadataStore"
                                default-cust-store="true">
            <metadata-store
class-name="oracle.mds.dt.persistence.stores.file.SrcControlFileMetadataStore">
              <property name="metadata-path" value="../../mds"/>
            </metadata-store>
          </metadata-store-usage>
        </metadata-store-usages>
      </persistence-config>
      <cust-config>
        <match>
          <customization-class name="oracle.adf.share.config.SiteCC"/>
        </match>
      </cust-config>
      <cache-config>
        <max-size-kb>100000</max-size-kb>
      </cache-config>

      <!-- Registration of a standalone XML file used for type-level customization
policies -->
      <type-config>
        <standalone-definitions>
          <file>Directory_Name/standalone.xml</file>
        </standalone-definitions>
      </type-config>
    </mds-config>
  </mdsC:adf-mds-config>

  <pe:page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">

    <!-- Composer add-on panel configuration -->
    <pe:addon-panels>
      <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-settings" />
      <pe:addon-panel name="oracle.adf.pageeditor.addonpanels.page-reset" />
      <pe:addon-panel name="oracle.fod.custom.panel" />
    </pe:addon-panels>

    <!-- Composer sandbox configuration -->
    <pe:sandbox-namespaces>
      <pe:namespace path="/pages"/>
      <pe:namespace path="/pageDefs"/>
    </pe:sandbox-namespaces>

    <!-- ComposerSessionOptionsFactory class registration for implementing
customization layers -->

<pe:session-options-factory>view.AppsSessionOptionsFactoryImpl</pe:session-options
-factory>

    <!-- Resource string editor configuration -->
    <pe:resource-string-editor>
      <pe:enabled>true</pe:enabled>
    </pe:resource-string-editor>
```

```
    <!-- Switch to enable or disable Composer Source view -->
    <pe:enable-source-view>false</pe:enable-source-view>

  </pe:page-editor-config>

  <!-- Multiple Resource Catalog configuration -->
  <rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
    <catalog-selector class-name="webcenter.CatalogSelector"/>
    <default-catalog catalog-name="users-catalog"/>
  </rcv-config>

  <!-- Composer and WebCenter Portal Customizable Components actions security
configuration -->
  <cust:customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
    <cust:enableSecurity value="true"/>
    <cust:actionsCategory>
      <cust:actionCategory name="personalizeActionsCategory" value="false"/>
      <cust:actionCategory name="editActionsCategory" value="true"/>
    </cust:actionsCategory>
    <cust:actions>
      <cust:action name="showMinimizeAction" value="true"/>
      <cust:action name="showMoveAction" value="false"/>
    </cust:actions>
  </cust:customizableComponentsSecurity>

  <adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/config">

    <!-- Composer persistence change manager configuration -->
      <persistent-change-manager>

<persistent-change-manager-class>oracle.adf.view.rich.change.MDSDocumentChangeMana
ger</persistent-change-manager-class>
      </persistent-change-manager>

    <!-- Composer default persistence configuration -->
    <taglib-config>
      <taglib uri="http://xmlns.oracle.com/adf/faces/customizable">
        <tag name="showDetailFrame">
          <persist-operations>all</persist-operations>
          <attribute name="expansionMode">
            <persist-changes>true</persist-changes>
          </attribute>
          <attribute name="contentStyle">
            <persist-changes>true</persist-changes>
          </attribute>
        </tag>
        <tag name="panelCustomizable">
          <persist-operations>all</persist-operations>
        </tag>
      </taglib>
    </taglib-config>
  </adf-faces-config>
</adf-config>
```

B.2.3 adf-settings.xml

The `adf-settings.xml` file, in addition to holding generic project- and library-level settings, can also be used to define custom actions on task flows. This section describes the `adf-settings.xml` configurations specific to Composer.

B.2.3.1 custComps-config

Use the `custComps-config` element to define custom actions on task flows consumed in the application. At runtime, the custom actions configurations in the application's `adf-config.xml` and `adf-settings.xml` files are picked up and appropriate actions are displayed on the `Show Detail Frame` component surrounding a task flow. For more information, see Section 22.3, "Enabling Custom Actions on a Task Flow."

Insert the `custComps-config` element within the `adf-settings` tag. Within the `custComps-config` element, you can specify the `customActions` element.

customActions

Use the `customActions` element in the `adf-settings.xml` file to define actions that you want to display on task flows. The custom actions defined here will be displayed along with the other actions on the surrounding `Show Detail Frame` component. You must add only one `customActions` element inside the `custComps-config` section. Within the `customActions` element you can add any number of `customAction` elements to define individual custom actions, as shown in the following example:

```
  <customActions>
   <customAction action="next" location="chrome"
                 rendered="true"
                 icon="/adf/webcenter/editheader_ena.png"
                 text="Next"

taskFlowId="/WEB-INF/task-flow-definition.xml#task-flow-definition"
                 shortDesc="next"/>
    <customAction action="prev" location="chrome"
                 rendered="true"
                 icon="/adf/webcenter/editheader_ena.png"
                 text="Previous"
 taskFlowId="/WEB-INF/task-flow-definition.xml#task-flow-definition"
                 shortDesc="prev"/>
  </customActions>
```

B.2.4 web.xml

The `web.xml` file is a Java EE standard descriptor that contains details about web applications. When you add a `Page Customizable` component to your page, the `web.xml` file available in the *`Application_Root`*`\`*`Project_Name`*`\public_html\WEB-INF` directory is updated to enable change persistence among other settings.

You must update `web.xml` further when creating an application that uses multiple customization layers and when enabling Composer sandbox. The following sections describe the `web.xml` configurations specific to Composer.

B.2.4.1 CHANGE_PERSISTENCE Context Parameter

The `CHANGE_PERSISTENCE` context parameter in the application's `web.xml` file specifies how application customizations are persisted. To enable changes in a

Composer-enabled page to be persisted in MDS, you must ensure that the `CHANGE_PERSISTENCE` context parameter is set to `ComposerChangeManager`, as shown in the following example:

```
<context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>

<param-value>oracle.adf.view.page.editor.change.ComposerChangeManager</param-value
>
</context-param>
```

This context parameter registers the `ChangeManager` class to be used to ensure persistence to MDS. This configuration happens automatically when you add a `Page Customizable` component to a page in a new Framework application. However, in an existing ADF application, this context parameter may be set to some other value, for example `session` or `oracle.adf.view.rich.change.FilteredPersistenceChangeManager`. Therefore, when you add Composer components to such an application, you must ensure that the `CHANGE_PERSISTENCE` parameter is set to `ComposerChangeManager`.

For more information, see Section 22.11, "Configuring the Persistence Change Manager."

B.2.4.2 WebCenterComposerFilter

Use `WebCenterComposerFilter` in the `web.xml` file to register Composer's `ComposerSessionOptionsFactory` with Oracle ADF for every HTTP request. The request is then handled depending on your implementation of `ComposerSessionOptionsFactory`. This filter can be defined as shown in the following example:

```
<filter-mapping>
<filter-name>WebCenterComposerFilter</filter-name>
  <url-pattern>/faces/*</url-pattern>
  <dispatcher>FORWARD</dispatcher>
  <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

You must use `WebCenterComposerFilter` if you are performing the following tasks:

- Creating customization layers. For information, see Section 23.3, "Adding Customization Layers to View and Edit Modes: Example."
- Enabling sandbox creation in Composer. For more information, see Section 23.2.1, "How to Enable Composer Sandbox Creation."

B.3 Composer Default Add-Ons and Property Panels

This section lists the add-ons and property panels available by default in Composer.

B.3.1 Default Add-Ons

This section lists the add-ons available in Composer and provides the names with which these add-ons are registered in the `adf-config.xml` file. It is useful to know these names as you will need them to enable or disable the display of the add-ons in your application.

- **Page Properties**

Used to edit page properties and create page parameters.

This add-on is registered with the name `oracle.adf.pageeditor.addonpanels.page-settings` in the `adf-config.xml` file.

- **Reset Page**

  Used to reset page customizations.

  This add-on is registered with the name `oracle.adf.pageeditor.addonpanels.page-reset` in the `adf-config.xml` file.

- **Customization Manager**

  Used to manage customizations made to pages and task flows.

  This add-on is registered with the name `oracle.adf.pageeditor.addonpanels.customization-manager` in the `adf-config.xml` file.

The Page Properties and Reset Page add-ons are rendered by default. If you create custom add-ons in your application, while registering those add-ons in `adf-config.xml`, you must also include the entries for the default add-ons. Without these, the default add-ons are not displayed in Composer. For information about add-ons, see Section 21.2, "Creating Composer Add-Ons."

B.3.2 Default Property Panels

This section lists the property panels available by default in the Component Properties and Page Properties dialogs and provides the names with which these panels are registered in the `pe_ext.xml` file. These names are useful if you choose to override the default panels with custom property panels, or hide some or all of the default panels. You can also enable specific panels for display in Composer Design view; for details, see Section 21.11, "Enabling Direct Select in Design View."

For information about creating property panels, see Section 21.3, "Creating Custom Property Panels."

- **Display Options**

  Used to define display-related behavior of a component. This panel is displayed in the Component Properties dialog for all components.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.generic-property-inspector`.

- **Display Options (Show Detail Frame)**

  Used to define display-related behavior of a `Show Detail Frame` component. This panel is displayed in the Component Properties dialog for a `Show Detail Frame` component only.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.sdfprop`.

- **Tabs**

  Used to define the display properties and order of tabs on a `Panel Customizable` or `Box` component. This panel is displayed in the Component Properties dialog for a `Panel Tabbed` or `Panel Customizable` component. The panel is displayed for a `Panel Customizable` component only if the component has tabs on it.

In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.panel-tabbed`.

- **Style**

  Used to define the appearance of the component instance. This panel is displayed in the Component Properties dialog for all components.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.inline-style-editor`.

- **Content Style**

  Used to define the appearance of content inside a component instance. This panel is displayed in the Component Properties dialog for all components.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.content-style-editor`.

- **Events**

  Used to wire a contextual event to an action handler to enable the passing of values from a producer component to a consumer component when the event is triggered on the producer. This panel displays in the Component Properties dialog for all components.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.events`.

- **Region Parameters**

  Used to define parameters for the task flow region. This panel displays in the Component Properties dialog only for task flow regions.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.region-param`.

- **Portlet Paramters**

  Used to define portlet parameters. This panel displays in the Component Properties dialog only for portlets.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.portlet-param`.

- **Layout Customizable**

  Used to define the layout for page components by selecting from a set of predefined layouts. This panel displays in the Component Properties dialog only for `Layout Customizable` components.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.layout-cust-prop`.

- **Child Components**

  Used to rearrange and show or hide child components inside a `Panel Customizable` component. This panel displays only for `Panel Customizable` components

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.adf.pageeditor.pane.child-navigator`.

- **Page Parameters**

  Used to define the page parameters that can be wired to components on the page. This panel displays in the Page Properties dialog.

In the `pe_ext.xml` file, this panel is referenced with the name `oracle.webcenter.page.pane.page-param`.

- **Page Security**

  Used to define page permissions. This panel displays in the Page Properties dialog for secured application pages.

  In the `pe_ext.xml` file, this panel is referenced with the name `oracle.webcenter.page.pane.page-sec`.

B.4 Composer Help Topic IDs

By default, context-sensitive help is not available for Composer panels and dialogs, and therefore Help icons are not displayed. These icons are displayed when you install a help provider and hook up help topics for the different panels and dialogs. Composer help topics are available in the `pages_cs.jar` file on the Oracle WebCenter Portal 11g Demonstrations and Samples page on the Oracle Technology Network (OTN) at:

`http://webcenter.oracle.com`

To integrate this help with the Composer UI in your Framework application, a help provider must be available to the application. After creating a help provider, use the topic IDs in Table B–17 to hook up individual help topics.

Table B–17 Composer Runtime Help Topics

Topic ID to be Used	Corresponding Composer UI
pages_cs_ pgeditmodesv	Source view of a page in Edit mode
pages_cs_ pgeditmodedv	Design view of a page in Edit mode
pages_cs_cpparams	Parameters tab in the Component Properties dialog
pages_cs_ cpdisplayopts	Display Options tab in the Component Properties dialog
pages_cs_ cptabs.html	Tabs tab in the Component Properties dialog
pages_cs_cpstyle	Style tab in the Component Properties dialog
pages_cs_cpcontstyle	Content Style tab in the Component Properties dialog
pages_cs_othercss	Other CSS option on the Style tab in the Component Properties dialog
pages_cs_cpevents	Events tab in the Component Properties dialog
pages_cs_compcat	Resource Catalog
pages_cs_ppgen	Display Options tab in the Page Properties dialog
pages_cs_ppsecurity	Security tab in the Component Properties dialog
pages_cs_ pppgparams	Parameters tab in the Component Properties dialog
pages_cs_ cplayoutprops	Layout Customizable tab in the Component Properties dialog for a `Layout Customizable` component
pages_cs_ resourceeditor	Select text resource dialog invoked on selecting the Select Text Resource option on the menu next to a field in the Component Properties dialog

Creating and Registering a Help Provider

This section describes an example where the help topics are extracted to a folder in the application Web context root and then hooked up to the UI elements.

> **Note:** For detailed information about creating a help provider and displaying help for components, see the section, "Displaying Help for Components" in the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

To create and register a help provider:

1. Download the help JAR from OTN and extract the help files to a folder in your application, for example, `Web Content/help`.
2. Create a provider class, for example, `WebCenterHelpProvider`, with the following code:

```
package portal;

import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;

import oracle.adf.view.rich.help.ResourceBundleHelpProvider;

public class WebCenterHelpProvider extends ResourceBundleHelpProvider {
    public WebCenterHelpProvider() {
        super();
    }

    public String getExternalUrl(FacesContext context,
                                       UIComponent component,
                                       String topicId) {
      return "http://localhost:7101/help_test-Portal-context-root/help/" +
topicId+".html";
    }
}
```

3. Register the new help provider in the `adf-settings.xml` file as follows:

```
<adf-settings xmlns="http://xmlns.oracle.com/adf/settings">
<adf-faces-config xmlns="http://xmlns.oracle.com/adf/faces/settings">
  <help-provider prefix="pages_cs">
    <help-provider-class>
      portal.WebCenterHelpProvider
    </help-provider-class>
    <property>
      <property-name>baseName</property-name>

<value>oracle.adfinternal.view.page.editor.resource.ComposerBundle</value>
    </property>
  </help-provider>
</adf-faces-config>
</adf-settings>
```

4. Run a page in your application and verify that the help shows up correctly.

B.5 Composer Components Style-Specific Properties

This section includes a series of tables that describe the style selectors associated with Composer components. Additionally, it describes how to use the `background` property to choose one of three skin-defined looks for these components.

This section contains the following subsections:

- Section B.5.1, "Style Selectors for Composer Components"
- Section B.5.2, "Style Attributes"

B.5.1 Style Selectors for Composer Components

Use a style selector to describe an element's appearance by identifying the element and defining styles for it.

This section contains the following subsections:

- Global Style Selectors
- Composer Status Indicator Style Selectors
- Page Customizable Style Selectors
- Layout Customizable Style Selectors
- Panel Customizable Style Selectors
- Show Detail Frame Style Selectors

B.5.1.1 Global Style Selectors

Use global style selectors to define styles for multiple components within the application. Table B–18 lists and describes the global style selectors relevant to Composer components.

Table B–18 Global Style Selectors

Style Selector	Description
`.ComposerDark:alias`	Specifies the default color for toolbars, headers, and so on in Composer, with a color scheme of *dark*.
`.ComposerLight:alias`	Specifies the default color for toolbars, headers, and so on in Composer, with a color scheme of *light*.
`.ComposerBackground`	Specifies the color scheme for Composer dialogs, such as the Component Properties and Page Properties dialogs.

Table B–18 (Cont.) Global Style Selectors

Style Selector	Description
`.ComposerBorder`	Specifies the border, width, and so on for Composer dialogs, such as the Component Properties and Page Properties dialogs.
`.PEClickableImageAnchor:alias`	An alias for rendering a clickable icon in a table cell. Specifies the CSS properties needed to display an image which is clickable and which has inline-mode display. For example, you can specify this as a common property for all action icons on a `Show Detail Frame` header. By default, this alias is included in style selectors that use 16x16 images and have to display inline background-images for both IE and Firefox. To customize, you can include this alias in your selector and override its properties. For example, you can change font-size and padding-right for an image with different height and width, or you can set "display: block" with height, width attributes to display in block mode.
`.ChildPanelFacetWarning`	Specifies the style for the warning message displayed in the child navigator in the Component Properties dialog.

B.5.1.2 Composer Status Indicator Style Selectors

Use the style selectors in Table B–19 to skin the status indicator component in the Composer toolbar.

Table B–19 Status Indicator Style Selectors

Style Selector	Description
`af|statusIndicator.ComposerStatus`	Specifies the style for the status indicator component. You can use this to specify a fixed height for the status indicator.
`af|statusIndicator.ComposerStatus::idle-icon`	Specifies the icon that represents the idle state. This icon is generally a non-animated icon that indicates communication is NOT occurring with the server.
`af|statusIndicator.ComposerStatus::processing-icon`	Specifies the icon that represents the processing state. This icon is generally an animated icon that indicates communication IS occurring with the server.
`af|statusIndicator.ComposerStatus::connecting-icon`	Specifies the icon that displays when the Active Data Service is in the process of connecting. This icon is generally an animated icon.
`af|statusIndicator.ComposerStatus::connected-icon`	Specifies the icon that displays when the Active Data Service is in the connected state. This icon is generally a non-animated icon and may be the same as the idle state icon.
`af|statusIndicator.ComposerStatus::reconnecting-icon`	Specifies the icon that displays when the Active Data Service has lost connection and is attempting to reconnect. This icon is generally an animated icon.
`af|statusIndicator.ComposerStatus::disconnected-icon`	Specifies the icon that displays when the Active Data Service is in the disconnected state. This icon is generally a non-animated icon used to indicate that no further reconnection attempts will be made without use interaction.

B.5.1.3 Page Customizable Style Selectors

Use the style selectors in Table B–20 to skin `Page Customizable` components.

Table B–20 Page Customizable Style Selectors

Style Selector	Description
af|pageCustomizable	Specifies the default size of a `Page Customizable` component when not being stretched by its parent container.
af|pageCustomizable af|toolbox.ComposerToolbox	Specifies the style for the main dark gray toolbar displayed in Edit mode. This toolbar contains a message about the page being edited and the various buttons.
af|pageCustomizable af|toolbox.ComposerToolbox af|panelGroupLayout.ComposerConcurrency	Specifies the style for the concurrency toolbar that appears below the main Composer toolbar when two or more users are editing a page at the same time in the same customization layer.
af|pageCustomizable af|toolbox.ComposerToolbox af|toolbar af|pageCustomizable af|toolbox.ComposerToolbox af|toolbar::separator af|pageCustomizable af|toolbox.ComposerToolbox af|toolbox::body af|pageCustomizable af|toolbox.ComposerToolbox af|toolbox::body af|toolbox::row af|pageCustomizable af|toolbox.ComposerToolbox af|toolbox::body af|toolbox::last-row af|pageCustomizable af|toolbox.ComposerToolbox af|toolbox::body af|toolbox::row af|toolbox::leading-cell	Specifies the styles for the different areas in the toolbar.
af|pageCustomizable af|toolbox.ComposerToolbox af|commandToolbarButton::text af|pageCustomizable af|toolbox.ComposerToolbox af|commandToolbarButton:hover af|pageCustomizable af|toolbox.ComposerToolbox af|commandToolbarButton:depressed	Specify the styles for the Composer toolbar buttons such as Page Properties and Reset Page.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar::body af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar::body:rtl	Specifies the style for the body of the View menu on the toolbar.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu::bar-item-text	Specifies the style for the text on the View menu.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu::bar-item:highlighted	Specifies the style for the View menu when the mouse hovers over it.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu::bar-item:depressed	Specifies the style for the View menu when it is clicked.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu::bar-item-open-icon-style	Specifies the style for the dropdown icon appearing on the View menu.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu:depressed af|menu::bar-item-open-icon-style	Specifies the style for the dropdown icon when it is clicked on the View menu.
af|pageCustomizable af|toolbox.ComposerToolbox af|menuBar af|menu:highlighted af|menu::bar-item-open-icon-style	Specifies the style for the dropdown icon when the mouse hovers over it.
af|pageCustomizable af|toolbar.SourceView	Specifies the style for the toolbar in Composer Source view.

Table B–20 (Cont.) Page Customizable Style Selectors

Style Selector	Description
af|pageCustomizable af|toolbar.SourceView af|menuBar	Specifies the style for the menubar in Source view.
af|dialog.ComposerDialog::content-start af|dialog.ComposerDialog::content-start:rtl af|dialog.ComposerDialog::content af|dialog.ComposerDialog::content:rtl af|dialog.ComposerDialog::content-end	Specify the styles for the dialogs invoked using Composer toolbar buttons. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.
af|pageCustomizable af|breadCrumbs.ComposerBreadcrumbs af|pageCustomizable af|breadCrumbs.ComposerBreadcrumbsHidden:step	Specify the styles for Composer breadcrumbs.
af|pageCustomizable:edit af|panelGroupLayout.ComposerSplitHorz af|pageCustomizable:edit af|panelGroupLayout.ComposerSplitVert	Specifies the style for `Panel Customizable` components that are split horizontally or vertically. Typically, these selectors are used to specify the height in case of a horizontal split, or width in case of a vertical split.
af|pageCustomizable:edit, af|pageCustomizable:stretched	Specifies the style for the `Page Customizable` component in Edit mode. This selector ensures that in Edit mode, a `Page Customizable` always stretches. In View mode, it will flow if being flowed, or stretch if being stretched.
af|pageCustomizable:printable	Specifies the style for the `Page Customizable` in printable mode. It suppresses the height and width of the `Page Customizable`.
af|pageCustomizable af|panelGroupLayout.ComposerCutComp af|tree::node-stamp-text	Specifies the style for the dotted line border around components that are cut.

B.5.1.4 Layout Customizable Style Selectors

Use the style selectors in Table B–21 to skin `Layout Customizable` components.

Table B–21 Layout Customizable Style Selectors

Style Selector	Description
af|layoutCustomizable::menu:edit	Specifies the style for the **Change Layout** button in Edit mode.
af|layoutCustomizable::menu:edit af|commandImageLink::text af|layoutCustomizable::menu:edit af|commandImageLink::text:hover	Specify the styles for the text on the **Change Layout** button in Edit mode.
af|layoutCustomizable.AFStretchWidth	Specifies the width to which the component can stretch horizontally.

B.5.1.5 Panel Customizable Style Selectors

Use the style selectors listed in Table B–22 to skin `Panel Customizable` components.

Note: Some style selectors in Table B–22 have three color-scheme selections: *light*, *medium*, and *dark*. Using these you can define three distinct looks in your CSS and specify which one to use through the `background` property in the JDeveloper Property Inspector. Depending on which value is specified for a component instance's `background` property, the skin applies the relevant style.

Table B–22 Panel Customizable Style Selectors

Style Selector	Description
`af|panelCustomizable`	Specifies the style for the root element of the component.
`af|panelCustomizable.PEStretched`	Specifies the style for the component, when it is stretched.
`af|panelCustomizable:edit`	Specifies the style for the container in Edit mode.
`af|panelCustomizable:edit:drop-target`	Specifies the style for this component when you drag another component on the page and hover over this component to drop it in.
`af|panelCustomizable::edit-mode-content-style`	Specifies the style for the area containing the Edit and Delete icons.

`af|panelCustomizable::add-icon-style`
`af|panelCustomizable::add-icon-style:active`
`af|panelCustomizable::add-icon-style:hover`
`af|panelCustomizable::add-icon-style:rtl`
	Specify the styles for the Add Content button on the component.

`af|panelCustomizable::edit-icon-style`
`af|panelCustomizable::edit-icon-style:active`
`af|panelCustomizable::edit-icon-style:hover`
	Specify the styles for the Edit icon on the component.

`af|panelCustomizable::delete-icon-style`
`af|panelCustomizable::delete-icon-style:active`
`af|panelCustomizable::delete-icon-style:hover`
	Specify the styles for the Delete icon on the component.

`.PanelCustomizableDropProxy`	Specifies the style for the placeholder area inside this component when you drag another component on the page and hover over this component to drop it in.

`af|panelCustomizable:edit:inline-selected`
`.p_AFActiveInlineEditableContainer`
`af|panelCustomizable:edit:hover-target`
	Specifies the styles for the components in the inline editable subtree in the `Panel Customizable` component.

`af|panelCustomizable::add-icon-link`
`af|panelCustomizable::add-icon-link:active`
`af|panelCustomizable::add-icon-link:hover`
	Specify the styles for the Add icon in the `Panel Customizable` component.

Table B–22 (Cont.) Panel Customizable Style Selectors

Style Selector	Description
af|panelCustomizable::addtab-icon-style af|panelCustomizable::addtab-icon-style:active af|panelCustomizable::addtab-icon-style:hover	Specify the styles for the Add Tab icon in the Panel Customizable component.
af|panelCustomizable::split-up-icon-style af|panelCustomizable::split-up-icon-style:active af|panelCustomizable::split-up-icon-style:hover af|panelCustomizable::split-down-icon-style af|panelCustomizable::split-down-icon-style:active af|panelCustomizable::split-down-icon-style:hover af|panelCustomizable::split-left-icon-style af|panelCustomizable::split-left-icon-style:active af|panelCustomizable::split-left-icon-style:hover af|panelCustomizable::split-right-icon-style af|panelCustomizable::split-right-icon-style:active af|panelCustomizable::split-right-icon-style:hover	Specify styles for the icons available on a Panel Customizable component to split it horizontally and vertically.
.PanelCustomizableDropProxy	Specifies the style for a component being dragged and dropped inside a Panel Customizable component.

B.5.1.6 Show Detail Frame Style Selectors

Use the style selectors listed in Table B–23 to skin portlets and Show Detail Frame components.

In Framework applications, each portlet is rendered with portlet *chrome* (for more information, see Section 58.1.1, "Portlet Anatomy"). The portlet chrome shares the same chrome rendering mechanism as a Show Detail Frame component. Thus, the style and icon selectors that apply to Show Detail Frame also apply to the portlet chrome. In other words, in addition to defining styles for Show Detail Frame components, use Show Detail Frame style and icon selectors to define styles for portlets.

Note: Some Show Detail Frame style selectors in Table B–23 have *light*, *medium*, *dark*, and *core:default* color scheme options. Using these you can define four distinct looks in your CSS and specify which one to use through the background property in the JDeveloper Property Inspector. Depending on which value is specified for a component instance's background property, the skin applies the relevant style.

Table B–23 Show Detail Frame Style Selectors

Style Selector	Description
af|showDetailFrame	Specifies the style for the root element of the component.
af|showDetailFrame::container:light af|showDetailFrame::container:medium af|showDetailFrame::container:dark af|showDetailFrame::container:core:default	Specify the styles for the element containing the contents within the component.
af|showDetailFrame.p_AFStretched	Specifies the style for the root element when the component is stretched.
af|showDetailFrame::content af|showDetailFrame::content:light af|showDetailFrame::content:medium af|showDetailFrame::content:dark af|showDetailFrame::content:core:default	Specify the styles to render for the content region of the component.
af|showDetailFrame::header af|showDetailFrame::header:light af|showDetailFrame::header:medium af|showDetailFrame::header:dark af|showDetailFrame::header:core:default	Specify the styles for the header element. This element surrounds the header text, icon, and actions regions.
af|showDetailFrame::header-center af|showDetailFrame::header-center:light af|showDetailFrame::header-center:medium af|showDetailFrame::header-center:dark af|showDetailFrame::header-center:core:default	Specify the styles for the region on the component header containing the icons, title, and actions menu.
af|showDetailFrame::header-start af|showDetailFrame::header-start:rtl af|showDetailFrame::header-start:light af|showDetailFrame::header-start:light:rtl af|showDetailFrame::header-start:medium af|showDetailFrame::header-start:medium:rtl af|showDetailFrame::header-start:dark af|showDetailFrame::header-start:dark:rtl af|showDetailFrame::header-start:core:default af|showDetailFrame::header-start:core:default:rtl	Specify the styles for the small cap-like area at the start of the header. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.
af|showDetailFrame::header-end af|showDetailFrame::header-end:rtl af|showDetailFrame::header-end:light af|showDetailFrame::header-end:light:rtl af|showDetailFrame::header-end:medium af|showDetailFrame::header-end:medium:rtl af|showDetailFrame::header-end:dark af|showDetailFrame::header-end:dark:rtl af|showDetailFrame::header-end:core:default af|showDetailFrame::header-end:core:default:rtl	Specify the styles for the small cap-like area at the end of the header. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.

Table B–23 (Cont.) Show Detail Frame Style Selectors

Style Selector	Description
af|showDetailFrame::footer-start af|showDetailFrame::footer-start:rtl af|showDetailFrame::footer-start:light af|showDetailFrame::footer-start:light:rtl af|showDetailFrame::footer-start:medium af|showDetailFrame::footer-start:medium:rtl af|showDetailFrame::footer-start:dark af|showDetailFrame::footer-start:dark:rtl af|showDetailFrame::footer-start:core:default af|showDetailFrame::footer-start:core:default:rtl	Specify the styles for the small cap-like area at the start of the footer. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.
af|showDetailFrame::footer-center af|showDetailFrame::footer-center:light af|showDetailFrame::footer-center:medium af|showDetailFrame::footer-center:dark af|showDetailFrame::footer-center:core:default	Specify the styles for the central region of the `Show Detail Frame` component's footer element.
af|showDetailFrame::footer-end af|showDetailFrame::footer-end:rtl af|showDetailFrame::footer-end:light af|showDetailFrame::footer-end:light:rtl af|showDetailFrame::footer-end:medium af|showDetailFrame::footer-end:medium:rtl af|showDetailFrame::footer-end:dark af|showDetailFrame::footer-end:dark:rtl af|showDetailFrame::footer-end:core:default af|showDetailFrame::footer-end:core:default:rtl	Specify the styles for the small cap-like area at the end of the footer. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.
af|showDetailFrame::shadowbox af|showDetailFrame::shadowbox:rtl af|showDetailFrame::shadowbox_div1 af|showDetailFrame::shadowbox_div1:rtl af|showDetailFrame::shadowbox_div2 af|showDetailFrame::shadowbox_div2:rtl	Specify styles for the shadow cast by the `Show Detail Frame` component.
af|showDetailFrame::toolbar af|showDetailFrame::toolbar:rtl	Specifies the styles for a floating toolbar on the `Show Detail Frame` component.
af|showDetailFrame::icon-style	Specifies the style for the icon on the `Show Detail Frame` header, if the `icon` attribute is set on the component.
af|showDetailFrame::header-text af|showDetailFrame::header-text:rtl	Specifies the style for the title available on the component header. Available alias is `:rtl`, which can be used when the component must be rendered right-to-left.
af|showDetailFrame::header-actions	Specifies the style for icons such as delete and the actions menu on the component header.
af|showDetailFrame::toolbar-container af|showDetailFrame::toolbar-container:light af|showDetailFrame::toolbar-container:medium af|showDetailFrame::toolbar-container:dark af|showDetailFrame::toolbar-container:core:default	Specify styles for the container element of the floating toolbar. Available aliases are `:light`, `:medium`, and `:dark`.

Table B–23 (Cont.) Show Detail Frame Style Selectors

Style Selector	Description
af|showDetailFrame::collapse-icon-style af|showDetailFrame::collapse-icon-style:active af|showDetailFrame::collapse-icon-style:hover af|showDetailFrame::collapse-icon-style.PEDisplayNone	Specify styles for the Collapse icon on the component header. Available aliases are `:active` and `:hover` that you can use to specify styles for the Collapse icon when clicked or when the mouse hovers over it. You can use `PEDisplayNone` to hide the Collapse icon.
af|showDetailFrame::disclose-icon-style af|showDetailFrame::disclose-icon-style:active af|showDetailFrame::disclose-icon-style:hover af|showDetailFrame::disclose-icon-style.PEDisplayNone	Specify the styles for the Disclose icon on the component header. You can use `PEDisplayNone` to hide the Disclose icon.
af|showDetailFrame::actionmenu-icon-style af|showDetailFrame::actionmenu-icon-style:active af|showDetailFrame::actionmenu-icon-style:hover	Specify the styles for the action menu icon. This style has a background-image that you can override.
af|showDetailFrame::remove-icon-style af|showDetailFrame::remove-icon-style:active af|showDetailFrame::remove-icon-style:hover	Specify the styles for the Delete icon.
af|showDetailFrame::edit-icon-style af|showDetailFrame::edit-icon-style:active af|showDetailFrame::edit-icon-style:hover	Specify the styles for the Edit icon.
af|showDetailFrame::preview-icon-style af|showDetailFrame::preview-icon-style:active af|showDetailFrame::preview-icon-style:hover	Specify styles for the Preview icon.
af|showDetailFrame::edit-mode-content-style af|showDetailFrame::edit-mode-content-style:light	Specify the styles for the area that contains the edit and remove icons.
af|showDetailFrame::edit-mode-edit-icon-style af|showDetailFrame::edit-mode-edit-icon-style:active af|showDetailFrame::edit-mode-edit-icon-style:hover	Specify the styles for the Edit icon on the component header in Edit mode.
af|showDetailFrame::edit-mode-remove-icon-style af|showDetailFrame::edit-mode-remove-icon-style:active af|showDetailFrame::edit-mode-remove-icon-style:hover	Specify the styles for the Delete icon on the component header in Edit mode.
af|showDetailFrame::edit-text-link af|showDetailFrame::edit-text-link:rtl af|showDetailFrame::edit-text-link:hover	Specify the styles for the Edit Text link in Edit mode when the component includes a Rich Text Editor component as a child.
af|showDetailFrame::resize af|showDetailFrame::resize:rtl	Specifies the style for the resize handler on the component. Available alias is `:rtl`, which can be used when component must be rendered right-to-left.

Table B–23 (Cont.) Show Detail Frame Style Selectors

Style Selector	Description
af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|toolbox > af|toolbox::body, af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|toolbox > af|toolbox::body > af|toolbox::row, af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|toolbox > af|toolbox::body > af|toolbox::last-row, af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|richTextEditor::content-input-container af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|richTextEditor::content-input, af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|richTextEditor::content-input-container af|showDetailFrame::content:medium.PEWrapsRTE af|showDetailFrame::content:light.PEWrapsRTE af|showDetailFrame::content.PEWrapsRTE > af|richTextEditor af|toolbox	Specify styles for the areas surrounding and inside a nested Rich Text Editor component. For example, to ensure that the background color of the Rich Text Editor component is transparent in Edit mode, or to provide borders around the component.
af|showDetailFrame::content.PEStretched	Specifies the height when the `Show Detail Frame` component stretches its content.
af|showDetailFrame::content.PEStretched:printable	Ensures that the `Show Detail Frame` does not restrict the height of its content in printable mode.
af|showDetailFrame::dynamic-help-icon-container af|showDetailFrame::dynamic-help-icon-container:rtl	Specifies the styles for the help icon container on the component header.
af|showDetailFrame::dynamic-help-icon-style af|showDetailFrame::dynamic-help-icon-style:active af|showDetailFrame::dynamic-help-icon-style:hover	Specify the styles for the help icon on the component header.
.PEShowDetailFrameAnimIcon	Specifies the animation icon to be used to show busy status on the component, for example, when you expand it.
.PEIconAndTextInHeading:alias .PEIconAndTextInHeading:alias:rtl	Specify the styles for rendering an icon with text in the component header.

B.5.2 Style Attributes

Table B–24 provides a quick reference for the style attributes available for WebCenter Portal Customizable Components and Composer components through the JDeveloper Property Inspector. Note that portlets also use these style attributes.

For detailed information about style properties, see the CSS Specifications page at:

http://www.w3.org/TR/CSS21/

Table B–24 Style Attributes of Composer and WebCenter Portal Customizable Components

Attribute	Description
`background-color`	The background color of the component. Choose from a list of colors, or click the Edit icon to select from a color palette.
`background-image`	The image that is displayed in the component background. Select to inherit from a parent component or to display no image, or click the Edit icon to select an image.
`background-repeat`	Specify whether and how the background image should repeat. Choose from: ■ `<none>`: Forgo repeating the image. ■ `no-repeat`: Forgo repeating the image. ■ `repeat`: Repeat the image to fill the container. ■ `repeat-x`: Repeat the image horizontally but not vertically. ■ `repeat-y`: Repeat the image vertically but not horizontally. ■ `inherit`: Inherit this setting from a parent component.
`border-color`	The color of the border that surrounds the component. Choose from a list of colors, or click the Edit icon to select from a color palette.
`border-style`	The style of border to draw around the component. Choose from: ■ `<none>`: Apply no style to the border. ■ `none`: Do not display a border (use `border-style:none` on the `ContentStyle` property to turn off portlet borders. Using this attribute with `InlineStyle` does not turn off portlet borders.) ■ `hidden`: Hide the border. ■ `dotted`: Display the border as a line of dots. ■ `double`: Display the border as a double line. ■ `groove`: Display the border as a grooved line. ■ `ridge`: Display the border as a ridged line. ■ `inset`: Display the border as a concave line. ■ `outset`: Display the border as a convex line. ■ `dashed`: Display the border as a line of dashes. ■ `solid`: Display the border as a solid line. ■ `inherit`: Inherit the border style from a parent component.
`border-width`	The thickness of the component border. Select `thick`, `medium`, `thin`, or `inherit` to determine the thickness of the border or inherit border width from a parent component. Alternatively, click the Edit icon and define a thickness in your preferred unit of measurement.
`color`	The color of the component text. Select from a list, or click the Edit icon to select from a color palette.
`font-family`	The font family (such as Arial, Helvetica, or sans-serif) for the component text. Enter this value manually.

Table B–24 (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components

Attribute	Description
font-size	The size of component text. Choose from: ■ Choose Length: Specify an absolute value for the font-size in your preferred unit of measurement. ■ Choose Percentage: Specify font size as a percentage of the normal text size defined for the browser. ■ inherit: Inherit font size from a parent component. ■ large: Select the next size larger than the font associated with the parent component. ■ larger: Increase the font size to larger than the font associated with the parent component (larger is larger than large). ■ medium: Set the font size to the default font size of the parent component. ■ small: Set the font size to smaller than the default font size of the parent component. ■ smaller: Set the font size to smaller than the default font size of the parent component (smaller is smaller than small).
font-style	The font style of the component text. Choose from: ■ <none>: Apply no style to the font. ■ normal: The font is not italic. ■ italic: The font is italic. ■ oblique: The font is slanted to look italic, though the font family may not have a formal italic member. ■ inherit: Inherit font style from a parent component.
font-weight	The thickness of the component text. Choose from: ■ <none>: Specify no weight for the component text. ■ normal: Apply same weight as the parent component's default font weight. ■ bold: Set component text in bold. ■ bolder: Set component text darker than bold. ■ light: Set component text lighter than normal. ■ lighter: Set component text lighter than light. ■ 100–900: Specify a value for text darkness—100 is lightest and 900 is darkest—400 is normal weight; 700 is bold. ■ inherit: Inherit font weight from a parent component.
height	The spacing to apply between lines of continuous text (also known as *leading*). Specify a height by choosing from: ■ Choose Length: Specify an absolute value in your preferred unit of measurement. ■ Choose Percentage: Specify a value as a percentage of the value set for a parent component. ■ auto: Set line height automatically, usually about 2 points larger than font size. ■ inherit: Inherit line height from the parent component.
list-style-image	An image to use as an indicator of a list item. Specify an image instead of defining a list-style-type. Choose from: ■ inherit: Inherit a list style image from a parent component. ■ none: Use no image as an indicator of a list item. Or click the Edit icon and select an image.

Table B–24 (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components

Attribute	Description
`list-style-type`	The type of indicator to use for items on a list. Use this instead of specifying a image through `list-style-image`. Select from among many styles offered on the list, including `<none>`, which means the browser's default style is used, and `inherit`, which means the `list-style-type` is inherited from a parent component.
`margin`	The border of space surrounding the component content. Choose from: ■ `Choose Length`: Specify an absolute value for all margins. ■ `Choose Percentage`: Set a value as a percentage of the margin of a parent component. ■ `inherit`: Inherit margin value from a parent component. ■ `auto`: Set the value automatically according to browser defaults.
`padding`	The amount of space between the component and its margin or, if there is a border, between the component and its border. Choose from: ■ `Choose Length`: Specify an absolute value in your preferred unit of measurement. ■ `Choose Percentage`: Specify the value as a percentage of the padding set for a parent component. ■ `inherit`: Inherit the value from a parent component.
`text-align`	The horizontal alignment of the component text. Choose from: ■ `<none>`: Use browser default. ■ `left` ■ `right` ■ `center` ■ `justify`: Align text so that all lines start and end at the same point left and right. ■ `auto`: Apply a value automatically, either left or right. ■ `inherit`: Inherit alignment from a parent component.

Table B–24 (Cont.) Style Attributes of Composer and WebCenter Portal Customizable Components

Attribute	Description
`text-decoration`	The decorative value to apply to component text. Choose from: ■ `<none>`: Use the browser default for hyperlinks; otherwise, no text decoration. ■ `none`: No text decoration, including no underscore for hyperlinks. ■ `underline`: Draw a line under all component text. ■ `overline`: Draw a line over all component text. ■ `line-through`: Draw a line through the middle of all component text. ■ `blink`: Make component text blink. ■ `inherit`: Inherit text decoration from a parent component.
`vertical-align`	The vertical alignment of component text. Choose from: ■ `Choose Length`: Specify an absolute value in your preferred unit of measurement. ■ `Choose Percentage`: Specify a value as a percentage of the value set for a parent component. ■ `inherit`: Inherit the value from a parent component. ■ `baseline`: Set the text on the baseline of the text row. ■ `middle`: Set the text in the middle of the text row. ■ `text-bottom`: Set the text on the bottom of the text row, lower than the baseline. ■ `text-top`: Set the text at the top of the text row, above the baseline. ■ `sub`: Set the text as subscript. ■ `super`: Set the text as superscript.
`width`	The spacing between text. Specify a width by choosing from: ■ `Choose Length`: Specify an absolute value in your preferred unit of measurement. ■ `Choose Percentage`: Specify a value as a percentage of the value set for a parent component. ■ `auto`: Set line height automatically, usually about 2 points larger than font size. ■ `inherit`: Inherit line height from the parent component.

B.6 Customizable Components (HTML) Properties

Customizable Components (HTML) are similar in function to customizable components from the 10.1.3.2.0 release and can be used to add more customizable components to your migrated 10.1.3.2.0 application page. These customizable components are available from the Customizable Components (HTML) tag library in JDeveloper. When you add customizable components to the page, default values are assigned to certain attributes and you can define values for the remaining attributes as required.

Customizable components enable users to minimize or maximize, hide or show, or move any component on the page at runtime.

Note: The Customizable Components (HTML) library is available only to support applications created in release 10.1.3.2.0. Adding these components to new applications is not supported.

It is recommended that you replace Customizable Components (HTML) in existing applications with their rich counterparts (from the Composer tag library) whenever possible.

B.6.1 Panel Customizable (HTML) Component

Use the `Panel Customizable (HTML)` component as a container for page content that can be customized at runtime. Components added inside a `Panel Customizable (HTML)` can be maximized, minimized, or rearranged.

Use this component only to perform runtime customizations on child components. If you just want a container to arrange components at design time, then it is recommended that you use an ADF Faces container like `Panel Group Layout`.

Table B–25 describes the attributes of a `Panel Customizable (HTML)` component.

Table B–25 Attributes of a Panel Customizable (HTML) Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ▪ Must not be a zero-length String. ▪ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ▪ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`.

Table B–25 (Cont.) Attributes of a Panel Customizable (HTML) Component

Attribute	Type	Supports EL?	Description
layout	List of values `horizontal/vertical`	Yes	Specifies how the children of the `Panel Customizable` (HTML) are laid out. The default value is `vertical`. Depending on the value of the layout attribute, child components are laid out as follows: If you select `vertical`, then the child components are displayed one below the other and can be moved either up or down within the layout. If you select `horizontal`, then the child components are displayed adjacent to each other and can be moved either to the left or right within the layout.
Style Attributes			
styleClass	String	Yes	A CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle`'s CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
Advanced Attributes			
binding	String	Yes	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean. For example: `binding="#{yourManagedBean.Binding}"`

B.6.2 Show Detail Frame (HTML) Component

Add `Show Detail Frame (HTML)` components inside `Panel Customizable (HTML)` components to enable user customizations like maximizing, moving, minimizing, restoring, and deleting child components. You can perform these customizations only if you add `Show Detail Frame (HTML)` components inside `Panel Customizable (HTML)` components.

Table B–26 describes the attributes of a `Show Detail Frame (HTML)` component.

Table B–26 Attributes of a Show Detail Frame (HTML) Component

Attribute	Type	Supports EL?	Description
Common Attributes			
id	String	No	The identifier for the component. The identifier must follow a subset of the syntax allowed in HTML: ▪ Must not be a zero-length String. ▪ First character must be an ASCII letter (A-Za-z) or an underscore ('_'). ▪ Subsequent characters must be an ASCII letter or digit (A-Za-z0-9), an underscore ('_'), or a dash ('-').
rendered	Boolean	Yes	Specifies whether the component is rendered. When set to `false`, no output is delivered for this component (the component is not in any way rendered, and cannot be made visible on the client). The default value is `true`.
text	String	Yes	A title for the `Show Detail Frame` (HTML) component.

icon	String	Yes	If you want to add an icon on the header of the `Show Detail Frame` component, then this specifies the URI for the image to be used.
			For example:
			`icon="http://source-pc/images/accessability.gif"`
			Note: An image that is stored at the document root does not require a full path. For example:
			`icon="detail.gif"`

Appearance Attributes			

Table B–26 (Cont.) Attributes of a Show Detail Frame (HTML) Component

Attribute	Type	Supports EL?	Description
shortDesc	String	Yes	The short description of the component. This text is commonly used by user agents to display tooltip help text, in which case the behavior for the tooltip is controlled by the user agent, for example, Firefox 2 truncates long tooltips. For form components, the `shortDesc` is displayed in a note window.
background	List of values `light/medium/dark`	Yes	Working with the skin CSS, provides a means of applying a different look and feel for this `Show Detail Frame (HTML)` instance. The default value is `medium`.
displayHeader	Boolean	Yes	Indicates whether the header of the `Show Detail Frame` (HTML) is displayed. The default value is `true`. If you set `displayHeader` to `false` and if you have exposed some actions on the component, then a toolbar is displayed when you move the mouse over the component area. The toolbar contains a drop down icon, which displays a menu of available options. This toolbar is rendered only if there are actions available on the component. The toolbar display is also affected by the `isSeededInteractionAvailable` attribute. As the default value for `isSeededInteractionAvailable` is `false`, the toolbar is not displayed. It can be displayed by setting `isSeededInteractionAvailable` to `true`.
expansionMode	List of values `maximized/minimized/normal`	Yes	The default state of the `Show Detail Frame (HTML)`. The default display mode is `normal`.
Actions Attributes			
showMoveAction	List of values `menu/none`	Yes	Specifies whether the Move action is displayed in the Actions menu. The default value is `menu`.
showMinimizeAction	List of values `chrome/none`	Yes	Specifies whether the minimize action is displayed on the header. The default is `chrome`.

Table B–26 (Cont.) Attributes of a Show Detail Frame (HTML) Component

Attribute	Type	Supports EL?	Description
showMaximizeAction	List of values `menu/none`	Yes	Specifies whether the minimize action is displayed on the header. The default value is `menu`.
Style Attributes			
contentStyle	String	Yes	The CSS style to apply to the `Show Detail Frame (HTML)` content area. Manually enter any style in compliance with CSS version 2.0 or later.
styleClass	String	Yes	A CSS style class to use for this component. The style class can be defined in your JSPX page or in a skinning CSS file, for example.
inlineStyle	String	Yes	The CSS styles to use for this component. This is intended for basic style changes. The `inlineStyle` is a set of CSS styles that are applied to the root DOM element of the component. If the `inlineStyle`'s CSS properties do not affect the DOM element you want affected, then you must create a skin and use the skinning keys which are meant to target particular DOM elements, like `::label` or `::icon-style`.
Behavior Attributes			
partialTriggers	String	Yes	The IDs of the components that should trigger a partial update. This component listens on the trigger components. If a trigger component receives an event that causes it to update in some way, this component requests to be updated too. Identifiers are relative to the source component (this component), and must account for NamingContainers. If your component is inside of a naming container, you can use a single colon to start the search from the root of the page, or multiple colons to move up through the NamingContainers - "`:`" pops out of the component's naming container (or itself if the component is a naming container) and begin the search from there, "`:::`" pops out of two naming containers (including itself if the component is a naming container) and begin the search from there.
disclosureListener	`javax.el.MethodExpression`	Yes	A method reference to a disclosure listener. A disclosure event is fired when the disclosure state changes.

Table B–26 (Cont.) Attributes of a Show Detail Frame (HTML) Component

Attribute	Type	Supports EL?	Description
Advanced Attributes			
binding	String	Yes	An EL reference that stores the component instance in a bean. This can be used to give programmatic access to a component from a backing bean, or to move creation of the component to a backing bean. For example: `binding="#{yourManagedBean.Binding}"`
attributeChangeListener	`javax.el.MethodExpression`	Yes	A method reference to an attribute change listener. Attribute change events are not delivered for any programmatic change to a property. They are only delivered when a renderer changes a property without the application's specific request. An example of an attribute change events might include the width of a column that supported client-side resizing.

The **Expression Builder** option available when setting these attributes enables you to bind the component instance to a managed bean property.

Show Detail Frame Facets

You can use the `Show Detail Frame (HTML)` facets to define and display custom actions on `Show Detail Frame (HTML)` components. Table B–27 describes the facets that enable you to display custom actions supported by the `Show Detail Frame (HTML)` component.

Table B–27 Show Detail Frame (HTML) Facets

Facet	Description
titleBarAction	Used if an action is to be associated with title of the `Show Detail Frame` (HTML) component.
additionalActions	Used if some additional actions are to be added to the list of actions available on the `Show Detail Frame (HTML)` header.

JDeveloper displays all facets available to the `Show Detail Frame (HTML)` component in the Structure window. However, only those that contain UI components appear activated.

B.6.3 Customizable Components (HTML) Style Selectors

The tables in this section describe the style selectors that you can use to skin WebCenter Portal Customizable Components.

This section contains the following subsections:

- Panel Customizable (HTML) Style Selectors

- Show Detail Frame (HTML) Style Selectors
- Property Keys
- Icon Selectors

B.6.3.1 Panel Customizable (HTML) Style Selectors

Use the style selectors listed in Table B–28 to skin `Panel Customizable (HTML)` components. For icon selectors relevant to the `Panel Customizable` component, see Section B.6.3.4, "Icon Selectors."

Table B–28 Panel Customizable (HTML) Style Selectors

Style Selector	Description
`afh|panelCustomizable::no-header-content`	Specifies the style to render for all four component borders when the component header is turned off.

B.6.3.2 Show Detail Frame (HTML) Style Selectors

Use the style selectors listed in Table B–29 to skin the `Show Detail Frame (HTML)` components.

> **Note:** In addition to defining styles for `Show Detail Frame` components, the `Show Detail Frame` style and icon selectors define styles for portlets.

Table B–29 Show Detail Frame (HTML) Style Selectors

Style Selector	Description

`afh|showDetailFrame::header-top-border-light`
`afh|showDetailFrame::header-top-border-medium`
`afh|showDetailFrame::header-top-border-dark`
	Specifies the style for the top and bottom border of a component header and, also the header background color.

`afh|showDetailFrame::container`	Specifies the style for the component's container.

`afh|showDetailFrame::header-light`
`afh|showDetailFrame::header-medium`
`afh|showDetailFrame::header-dark`
	Specifies the style for text in the component's header. The header is usually a banner of color that contains a title and links to menus and other types of actions.
	Define the header background color with the `af|showDetailFrame::header-top-border-[light,medium,dark]` style element. Use icon selectors to specify the icons to use in the component header and, also the shape of the header's left and right sides.

`afh|showDetailFrame::content-light`
`afh|showDetailFrame::content-medium`
`afh|showDetailFrame::content-dar`
	Specifies the style for the component's left, right, and bottom borders.

`afh|showDetailFrame::main-menu-container`	Specifies the style for the component's main menu container.
`afh|showDetailFrame::sub-menu-container`	Specifies the style for the component's submenu container.
`afh|showDetailFrame::menu-item`	Specifies the style for an individual item on the component's main menu.
`afh|showDetailFrame::menu-item:hover`	Specifies the style to render when the mouse hovers over a component main menu item.
`afh|showDetailFrame::sub-menu-item`	Specifies the style for an individual item on the component's submenu.

Table B–29 (Cont.) Show Detail Frame (HTML) Style Selectors

Style Selector	Description
`afh|showDetailFrame::sub-menu-item:hover`	Specifies the style to render when the mouse hovers over a component submenu item.
`afh|showDetailFrame::actions-image-separator`	Specifies the amount of padding to provide around the component's Actions, Minimize, and Restore icons. For more information, see also Section B.6.3.4, "Icon Selectors".
`afh|showDetailFrame::menu-item-separator`	Specifies the style for the line that separates a command or groups of commands on the component's Actions menu. In the default case, a separator appears to be a single thick line. This is achieved using `border-top` and `border-bottom` elements to style the separator. A user who creates a custom skin can style the separator differently. For example, a user can create a separator that displays as a rectangular bar with a colorful background.
`A.afh|showDetailFrame::title-clickable`	Specifies the style to render for the component's title when the title is a link.

`afh|showDetailFrame::no-header-content`
`afh|showDetailFrame::no-header-content-light`
`afh|showDetailFrame::no-header-content-medium`
`afh|showDetailFrame::no-header-content-dark`
	Specifies the style to render for all four component borders when the component header is turned off.

B.6.3.3 Property Keys

Use property keys to control the display of custom menu items and component action icons. Though you include property keys in your custom skin, they are not represented in the generated CSS that results from the skin.

To explain, skins go through a process that results in a generated CSS. In turn, the generated CSS is consumed by the application. Most style selectors for `Panel Customizable` and `Show Detail Frame` components are represented in the generated CSS. Property keys are the exception. Although they affect the application as much as any other component style selector, they are not represented in the final generated CSS.

Table B–30 lists and describes property keys relevant to `Panel Customizable` and `Show Detail Frame` components.

Table B–30 Property Keys of Panel Customizable and Show Detail Frame Components

Property Key	Description
`showDetailFrame {-ora-additional-actions-position-last:true}`	Positions additional actions relative to seeded actions on the component's Actions menu. Set to `false` to position additional actions before seeded actions. By default, additional actions are positioned after seeded actions. The default value is `true`.
`showDetailFrame {-ora-menu-icon-display:false}`	Controls the display of icons next to their related commands on a `Show Detail Frame` Actions menu. Set to `true` to display icons to the left of each action on the Actions menu. By default, no icons are displayed to the left of individual actions. The default value is `false`. For information about specifying the icons to use when this property key is set to `true`, see Section B.6.3.4, "Icon Selectors."

B.6.3.4 Icon Selectors

Icons are either displayed or not displayed depending on whether the component's `ora-menu-icon-display` property key is set to `true` or `false`. Property keys are described in Section B.6.3.3, "Property Keys."

Each icon selector has a *light*, *medium*, and *dark* scheme. Using these you can define three distinct looks in your CSS and specify which one to use through the `background` property in the JDeveloper Property Inspector. Depending on which value is specified for a component instance's `background` property, the skin applies the relevant style.

For easy, error-free portability, store all application icons under the Framework application's root folder.

The selectors described in Table B–31 apply to the icons used with `Panel Customizable (HTML)` and `Show Detail Frame (HTML)` components.

Table B–31 Icon Selectors for WebCenter Portal Customizable Components

Selector	Description
showDetailFrame showDetailFrame::light-ActionsIcon:alias showDetailFrame::medium-ActionsIcon:alias showDetailFrame::dark-ActionsIcon:alias **panelCustomizable** panelCustomizable::light-ActionsIcon:alias panelCustomizable::medium-ActionsIcon:alias panelCustomizable::dark-ActionsIcon:alias	This icon represents the Actions menu. The Actions menu lists the actions a user can perform on the component. In a Framework application, the Actions icon is rendered on the right corner of the component header.
showDetailFrame showDetailFrame::light-MinimizeIcon:alias showDetailFrame::medium-MinimizeIcon:alias showDetailFrame::dark-MinimizeIcon:alias **panelCustomizable** panelCustomizable::light-MinimizeIcon:alias panelCustomizable::medium-MinimizeIcon:alias panelCustomizable::dark-MinimizeIcon:alias	This icon represents the Minimize option. Minimize collapses the view of the component like a window shade. In a Framework application, the Minimize icon is rendered on the left side of the component header. See also, showDetailFrame::light-ExpandIcon:alias.
showDetailFrame showDetailFrame::light-MaximizeIcon:alias showDetailFrame::medium-MaximizeIcon:alias showDetailFrame::dark-MaximizeIcon:alias	This is applicable only for the Show Detail Frame (HTML) component. The rich variant of this component does not support the maximize action. This icon represents the Maximize option, which expands the Show Detail Frame (HTML) component to the dimensions of the Panel Customizable (HTML) component that contains it. Where multiple Show Detail Frame (HTML) components display in the same container, these are displaced while the maximized Show Detail Frame (HTML) remains maximized. In a Framework application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.
panelCustomizable panelCustomizable::light-MaximizeIcon:alias panelCustomizable::medium-MaximizeIcon:alias panelCustomizable::dark-MaximizeIcon:alias	This is applicable only for the Panel Customizable (HTML) component. The rich variant of this component does not provide any actions. This icon represents the Maximize option, which expands component display to the dimensions of the container. Where multiple components display in the same container, these are displaced by the maximized component. In a Framework application, the Maximize icon is displayed to the left of the Maximize command on the component's Actions menu.
showDetailFrame showDetailFrame::light-RestoreIcon:alias showDetailFrame::medium-RestoreIcon:alias showDetailFrame::dark-RestoreIcon:alias **panelCustomizable** panelCustomizable::light-RestoreIcon:alias panelCustomizable::medium-RestoreIcon:alias panelCustomizable::dark-RestoreIcon:alias	This icon represents the Restore option, which restores maximized views to their default display modes. In a Framework application, the Restore icon is rendered to the left of the Restore command on the component's Actions menu.

Table B–31 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

Selector	Description
showDetailFrame showDetailFrame::light-ExpandIcon:alias showDetailFrame::medium-ExpandIcon:alias showDetailFrame::dark-ExpandIcon:alias **panelCustomizable** panelCustomizable::light-ExpandIcon:alias panelCustomizable::medium-ExpandIcon:alias panelCustomizable::dark-ExpandIcon:alias	The Expand icon represents the action that expands a component that has been minimized. The Expand icon toggles with the Minimize icon. That is, when the component is minimized, the Expand icon is displayed; when the component is expanded, the Minimize icon is displayed. In a Framework application, the Expand icon is displayed on the left side of the component header.
showDetailFrame showDetailFrame::light-MoveIcon:alias showDetailFrame::medium-MoveIcon:alias showDetailFrame::dark-MoveIcon:alias **panelCustomizable** panelCustomizable::light-MoveIcon:alias panelCustomizable::medium-MoveIcon:alias panelCustomizable::dark-MoveIcon:alias	This icon represents the Move option, which enables rearrangement of a component's location in relation to the other components on the page. In a Framework application, the Move icon is displayed to the left of the Move command on the component's Actions menu.
showDetailFrame showDetailFrame::light-MoveLeftIcon:alias showDetailFrame::medium-MoveLeftIcon:alias showDetailFrame::dark-MoveLeftIcon:alias **panelCustomizable** panelCustomizable::light-MoveLeftIcon:alias panelCustomizable::medium-MoveLeftIcon:alias panelCustomizable::dark-MoveLeftIcon:alias	This icon represents the Move Left option on the component submenu. Move Left rearranges the component horizontally, one position closer to the left boundary of the page. For example, imagine three horizontally arranged components. You select Move Left on the right-most component. It becomes the middle component. In a Framework application, the Move Left icon is displayed to the left of the Move Left submenu item on the component's Actions menu.
showDetailFrame showDetailFrame::light-MoveRightIcon:alias showDetailFrame::medium-MoveRightIcon:alias showDetailFrame::dark-MoveRightIcon:alias **panelCustomizable** panelCustomizable::light-MoveRightIcon:alias panelCustomizable::medium-MoveRightIcon:alias panelCustomizable::dark-MoveRightIcon:alias	This icon represents the Move Right option on the component submenu. Move Right rearranges the component horizontally, one position closer to the right boundary of the page. For example, imagine three horizontally arranged components. You select Move Right on the left-most component. It becomes the middle component. In a Framework application, the Move Right icon is displayed to the left of the Move Right submenu item on the component's Actions menu.
showDetailFrame showDetailFrame::light-MoveUpIcon:alias showDetailFrame::medium-MoveUpIcon:alias showDetailFrame::dark-MoveUpIcon:alias **panelCustomizable** panelCustomizable::light-MoveUpIcon:alias panelCustomizable::medium-MoveUpIcon:alias panelCustomizable::dark-MoveUpIcon:alias	This icon represents the Move Up option on the component submenu. Move Up rearranges the component vertically in relation to the other components on the page. For example, imagine three vertically arranged components. You select Move Up on the middle component. It becomes the topmost component. In a Framework application, the Move Up icon is displayed to the left of the Move Up submenu item on the component's Actions menu.

Table B–31 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

Selector	Description
showDetailFrame showDetailFrame::light-MoveDownIcon:alias showDetailFrame::medium-MoveDownIcon:alias showDetailFrame::dark-MoveDownIcon:alias **panelCustomizable** panelCustomizable::light-MoveDownIcon:alias panelCustomizable::medium-MoveDownIcon:alias panelCustomizable::dark-MoveDownIcon:alias	This icon represents the Move Down option on the component submenu. Move Down rearranges the component vertically in relation to the other components on the page. For example, imagine three vertically arranged components. You select Move Down on the middle component. It becomes the bottommost component. In a Framework application, the Move Down icon is displayed to the left of the Move Down submenu item on the component's Actions menu.
showDetailFrame showDetailFrame::light-HeaderLeftIcon:alias showDetailFrame::medium-HeaderLeftIcon:alias showDetailFrame::dark-HeaderLeftIcon:alias **panelCustomizable** panelCustomizable::light-HeaderLeftIcon:alias panelCustomizable::medium-HeaderLeftIcon:alias panelCustomizable::dark-HeaderLeftIcon:alias	This icon provides an image for the top-left corner of the component header.
showDetailFrame showDetailFrame::light-HeaderRightIcon:alias showDetailFrame::medium-HeaderRightIcon:alias showDetailFrame::dark-HeaderRightIcon:alias **panelCustomizable** panelCustomizable::light-HeaderRightIcon:alias panelCustomizable::medium-HeaderRightIcon:alias panelCustomizable::dark-HeaderRightIcon:alias	This icon provides the image for the top-right corner of the component header.
showDetailFrame showDetailFrame::light-ToolbarLeftIcon:alias showDetailFrame::medium-ToolbarLeftIcon:alias showDetailFrame::dark-ToolbarLeftIcon:alias **panelCustomizable** panelCustomizable::light-ToolbarLeftIcon:alias panelCustomizable::medium-ToolbarLeftIcon:alias panelCustomizable::dark-ToolbarLeftIcon:alias	This icon provides the left portion of the component's FadeIn-FadeOut toolbar. The FadeIn-FadeOut toolbar comes into play when the `adfp:portlet` tag attribute `isSeededInteractionAvailable` is set to `true` and `displayHeader` is set to `false`. The toolbar contains the Actions menu that would otherwise be displayed on the header. To invoke the toolbar, users move their mouse over the component content area. If the page design is very simple, then the FadeIn-FadeOut toolbar may not display, even when `displayHeader` is set to `false` and `isSeededInteractionAvailable` is set to `true`.
showDetailFrame showDetailFrame::light-ToolbarRightIcon:alias showDetailFrame::medium-ToolbarRightIcon:alias showDetailFrame::dark-ToolbarRightIcon:alias **panelCustomizable** panelCustomizable::light-ToolbarRightIcon:alias panelCustomizable::medium-ToolbarRightIcon:alias panelCustomizable::dark-ToolbarRightIcon:alias	This icon provides the right portion of the component's FadeIn-FadeOut toolbar. See the description for `showDetailFrame::light-ToolbarLeftIcon:alias` for an explanation of the FadeIn-FadeOut toolbar.

Table B–31 (Cont.) Icon Selectors for WebCenter Portal Customizable Components

Selector	Description
showDetailFrame showDetailFrame::light-ToolbarCenterIcon:alias showDetailFrame::medium-ToolbarCenterIcon:alias showDetailFrame::dark-ToolbarCenterIcon:alias **panelCustomizable** panelCustomizable::light-ToolbarCenterIcon:alias panelCustomizable::medium-ToolbarCenterIcon:alias panelCustomizable::dark-ToolbarCenterIcon:alias	This icon provides the center portion of the component's FadeIn-FadeOut toolbar. See the description for `showDetailFrame::light-ToolbarLeftIcon:alias` for an explanation of the FadeIn-FadeOut toolbar.
showDetailFrame showDetailFrame::light-EditIcon:alias showDetailFrame::medium-EditIcon:alias showDetailFrame::dark-EditIcon:alias **panelCustomizable** panelCustomizable::light-EditIcon:alias panelCustomizable::medium-EditIcon:alias panelCustomizable::dark-EditIcon:alias	This icon represents the Edit option on the component menu. In a Framework application, the Edit icon is displayed to the left of the Edit menu item on the component's Actions menu.
showDetailFrame showDetailFrame::light-HelpIcon:alias showDetailFrame::medium-HelpIcon:alias showDetailFrame::dark-HelpIcon:alias **panelCustomizable** panelCustomizable::light-HelpIcon:alias panelCustomizable::medium-HelpIcon:alias panelCustomizable::dark-HelpIcon:alias	This icon represents the Help option on the component menu. In a Framework application, the Help icon is displayed to the left of the Help menu item on the component's Actions menu.

C

Resource Catalog Properties and Files

This appendix provides reference information about the configuration, location, and attributes of the default catalog definition file. This information is useful while performing the tasks described in Part III, "Customizing Your Application and Extending Customization Options".

For detailed information about Resource Catalogs, see Chapter 16, "Creating and Managing Resource Catalogs."

This appendix includes the following sections:

- Section C.1, "Configuration and Location of Catalog Definitions"
- Section C.2, "XML Schema"
- Section C.3, "Catalog Definition Attributes"
- Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog"

C.1 Configuration and Location of Catalog Definitions

The `rcv-config` section of your `adf-config.xml` file defines the name of the default catalog. The `rcv-config` element is not available by default in your application's `adf-config.xml` file. You must add it if you want to change the name of the default catalog or, if you have chosen to implement multiple catalogs, specify a catalog selector class.

```
<rcv:rcv-config>
  <rcv:catalog-selector class-name=""/>
  <rcv:default-catalog
catalog-name="/oracle/webcenter/portalapp/catalogs/default-catalog.xml"/>
</rcv:rcv-config>
```

The Resource Catalog service stores catalog definition files in MDS at the *`resource-catalog-root`* directory. The *`resource-catalog-root`* is defined as an MDS namespace, hence the exact location depends on your MDS configuration. The default *`resource-catalog-root`* is defined by the `default-scope` attribute in the `adf-rcs-config` element in your `adf-config.xml`. This attribute is set to "**`/`**" implying that a catalog ID is expressed as fully the qualified MDS path:

```
<rcs:adf-rcs-config>
  <rcs:rcs-config>
    <rcs:catalog-config default-scope="/"

default-registry="/oracle/webcenter/portalapp/catalogs/catalog-registry.xml"/>
    . . .
```

```
    <rcs:security-manager
class-name="oracle.webcenter.portalframework.genericsiteresources.internal.securit
y.CatalogSecurityManager"/>
  </rcs:rcs-config>
```

The `security-manager` element defines the security manager implementation.

C.2 XML Schema

Example C–1 provides the catalog definition XML schema for your reference.

Example C–1 XML Schema for Catalog Definition

```
<xsd:schema elementFormDefault="qualified"
            targetNamespace="http://xmlns.oracle.com/adf/rcs/catalog"
            xmlns:rcs="http://xmlns.oracle.com/adf/rcs/catalog"
            xmlns:md="http://xmlns.oracle.com/bali/xml/metadata"
            xmlns:mds="http://xmlns.oracle.com/mds"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:jaxb="http://java.sun.com/xml/ns/jaxb" jaxb:version="1.0">

  <xsd:annotation>
    <xsd:appinfo>
      <jaxb:schemaBindings>
        <jaxb:package name="oracle.adfinternal.rc.catalog.xml"/>
      </jaxb:schemaBindings>
    </xsd:appinfo>
  </xsd:annotation>

  <!-- declare all elements as top-level elements so they appear in the
       component palette pages generated by XMLEF -->

  <!-- IMPORTANT: catalogDefinition & navigationModel CANNOT share a common type
-->
  <!--            MDS cannot handle it. -->
  <xsd:element name="catalogDefinition" type="rcs:catalogDefinitionType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <!-- enable customization for navigation models -->
  <xsd:element name="navigationDefinition" type="rcs:navigationDefinitionType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <!-- disable customization of the schema definition for both catalog &
navigation models -->
  <xsd:element name="schema" type="rcs:schemaType">
    <xsd:annotation>
```

```
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="descriptor" type="rcs:descriptorType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="attributes" type="rcs:attributesType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="attribute" type="rcs:attributeType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
          <!-- ### this marks "attributeId" attribute as "locally unique" within
the <attributes> tag -->
          <mds:localUniqueAttribute>attributeId</mds:localUniqueAttribute>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="parameters" type="rcs:parametersType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
        </md:elementMetadata>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="parameter" type="rcs:parameterType">
    <xsd:annotation>
      <xsd:appinfo>
        <md:elementMetadata>
          <mds:customizationAllowed>true</mds:customizationAllowed>
          <!-- ### this marks "id" attribute as "locally unique" within the
<parameters> tag -->
          <mds:localUniqueAttribute>id</mds:localUniqueAttribute>
        </md:elementMetadata>
```

```
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="contents" type="rcs:contentsType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="folder" type="rcs:folderType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="resource" type="rcs:resourceType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="includeCatalog" type="rcs:includeCatalogType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="includeNavigation" type="rcs:includeNavigationType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="repository" type="rcs:repositoryType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
```

```
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="dynamicFolder" type="rcs:dynamicFolderType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="source" type="rcs:dynamicFolderSourceType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="customFolder" type="rcs:customFolderType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="component" type="rcs:componentType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="url" type="rcs:urlType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="customContent" type="rcs:customContentType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
```

```
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>

<xsd:element name="separator" type="rcs:separatorType">
  <xsd:annotation>
    <xsd:appinfo>
      <md:elementMetadata>
        <mds:customizationAllowed>true</mds:customizationAllowed>
      </md:elementMetadata>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>



<xsd:complexType name="descriptorType">
  <xsd:attribute name="attributeId" type="xsd:string" use="required"/>
  <xsd:attribute name="labelKey" type="xsd:string"/>
  <xsd:attribute name="shortLabelKey" type="xsd:string"/>
  <xsd:attribute name="helpTextKey" type="xsd:string"/>
  <xsd:attribute name="hintTextKey" type="xsd:string"/>
  <xsd:attribute name="searchable" type="xsd:boolean" default="true"/>
  <xsd:attribute name="multivalue" type="xsd:boolean" default="false"/>
  <xsd:attribute name="endUserVisible" type="xsd:boolean" default="true"/>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>


<xsd:complexType name="schemaType">
  <xsd:sequence>
    <xsd:element ref="rcs:descriptor" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>


<xsd:complexType name="attributeType">
  <xsd:attribute name="attributeId" type="xsd:string" use="required"/>
  <xsd:attribute name="value" type="xsd:string" use="required"/>
  <xsd:attribute name="isKey" type="xsd:boolean" default="false"/>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>

<xsd:complexType name="attributesType">
  <xsd:sequence>
    <xsd:element ref="rcs:attribute" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="resourceBundle" type="xsd:string"/>
</xsd:complexType>

<xsd:complexType name="parameterType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="id" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

<xsd:complexType name="parametersType">
```

```
    <xsd:sequence>
      <xsd:element ref="rcs:parameter" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="itemType">
    <xsd:all>
      <xsd:element ref="rcs:attributes" minOccurs="0" maxOccurs="1"/>
      <xsd:element ref="rcs:parameters" minOccurs="0" maxOccurs="1"/>
    </xsd:all>
    <xsd:attribute name="id" type="xsd:ID" use="required"/>
    <xsd:attribute name="name" type="xsd:string"/>
    <xsd:attribute name="description" type="xsd:string"/>
    <xsd:attribute name="visible" type="xsd:string" default="true"/>
  </xsd:complexType>


  <xsd:complexType name="includeCatalogType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="scope" type="xsd:string" use="required"/>
        <xsd:attribute name="catalogId" type="xsd:string" use="required"/>
        <xsd:attribute name="insertFolderContents" type="xsd:boolean"
default="false"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="includeNavigationType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="scope" type="xsd:string" use="required"/>
        <xsd:attribute name="navigationId" type="xsd:string" use="required"/>
        <xsd:attribute name="insertFolderContents" type="xsd:boolean"
default="false"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="repositoryType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="repository" type="xsd:string" use="required"/>
        <xsd:attribute name="path" type="xsd:string"/>
        <xsd:attribute name="includeSubfolders" type="xsd:boolean"
default="true"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>


  <xsd:complexType name="dynamicFolderSourceType">
    <xsd:attribute name="repository" type="xsd:string"/>
    <xsd:attribute name="path" type="xsd:string"/>
    <xsd:attribute name="includeSubfolders" type="xsd:boolean" default="true"/>
  </xsd:complexType>

  <xsd:complexType name="dynamicFolderType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
```

```
        <xsd:sequence>
          <xsd:element ref="rcs:source" minOccurs="1" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="expression" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="customContentType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="contentProviderClass" type="xsd:string"
use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="customFolderType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="factoryClass" type="xsd:string" use="required"/>
        <xsd:attribute name="path" type="xsd:string"/>
        <xsd:attribute name="insertFolderContents" type="xsd:boolean"
default="false"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- this is a bit of a kludge. To keep MDS happy we must define a type -->
  <!-- for separators even though they don't have any special attributes -->
  <xsd:complexType name="separatorType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType"/>
    </xsd:complexContent>
  </xsd:complexType>

<!-- implementation of filters has been deferred

  <xsd:complexType name="filterType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:attribute name="expression" type="xsd:string"/>
        <xsd:attribute name="filterClass" type="xsd:string"/>
        <xsd:attribute name="scope" type="xsd:string" default="subtree"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
-->

  <xsd:group name="folderContentGroup">
    <xsd:choice>
      <xsd:element name="folder" type="rcs:folderType"/>
      <xsd:element name="repository" type="rcs:repositoryType"/>
      <xsd:element name="resource" type="rcs:resourceType" />
      <xsd:element name="dynamicFolder" type="rcs:dynamicFolderType" />
      <xsd:element name="customFolder" type="rcs:customFolderType"/>
      <xsd:element name="customContent" type="rcs:customContentType"/>
      <xsd:element name="component" type="rcs:componentType"/>
      <xsd:element name="url" type="rcs:urlType"/>
      <xsd:element name="includeCatalog" type="rcs:includeCatalogType"/>
```

```
      <xsd:element name="includeNavigation" type="rcs:includeNavigationType"/>
      <xsd:element name="separator" type="rcs:separatorType"/>
    </xsd:choice>
  </xsd:group>

   <xsd:complexType name="contentsType">
      <xsd:group ref = "rcs:folderContentGroup" minOccurs="0"
maxOccurs="unbounded" />
   </xsd:complexType>

  <!-- contents is optional to allow resourceType to extend folderType -->
  <!-- most resources won't have children so the contents element should be
optional -->
  <xsd:complexType name="folderType">
    <xsd:complexContent>
      <xsd:extension base="rcs:itemType">
        <xsd:all>
<!--      <xsd:element name="filter" type="rcs:filterType" minOccurs="0"
maxOccurs="unbounded"/> -->
          <xsd:element ref="rcs:contents" minOccurs="0" maxOccurs="1"/>
        </xsd:all>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- resourceType extends folderType to allow resource elements to have children
-->
  <xsd:complexType name="resourceType">
    <xsd:complexContent>
      <xsd:extension base="rcs:folderType">
        <xsd:attribute name="repository" type="xsd:string" use="required"/>
        <xsd:attribute name="path" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- resourceType extends folderType to allow component elements to have
children -->
  <xsd:complexType name="componentType">
    <xsd:complexContent>
      <xsd:extension base="rcs:folderType">
        <xsd:attribute name="factoryClass" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- resourceType extends folderType to allow URL elements to have children -->
  <xsd:complexType name="urlType">
    <xsd:complexContent>
      <xsd:extension base="rcs:folderType">
        <xsd:attribute name="url" type="xsd:string" use="required"/>
        <xsd:attribute name="factoryClass" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="catalogDefinitionType">
    <xsd:complexContent>
      <xsd:extension base="rcs:folderType">
        <xsd:all>
```

```
          <xsd:element ref="rcs:schema" minOccurs="0" maxOccurs="1"/>
        </xsd:all>
        <xsd:attribute name="contact" type="xsd:string"/>
        <xsd:attribute name="definitionFilter" type="xsd:string"/>
        <xsd:attribute name="resourceBundle" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="navigationDefinitionType">
    <xsd:complexContent>
      <xsd:extension base="rcs:folderType">
        <xsd:all>
          <xsd:element ref="rcs:schema" minOccurs="0" maxOccurs="1"/>
        </xsd:all>
        <xsd:attribute name="contact" type="xsd:string"/>
        <xsd:attribute name="definitionFilter" type="xsd:string"/>
        <xsd:attribute name="resourceBundle" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

</xsd:schema>
```

C.3 Catalog Definition Attributes

The catalog definition schema provides a rich set of elements through which you can specify the structure and content of your Resource Catalogs.

This section contains the following subsections:

- Section C.3.1, "catalogDefinition"
- Section C.3.2, "folder"
- Section C.3.3, "url"
- Section C.3.4, "includeCatalog"
- Section C.3.5, "component"
- Section C.3.6, "customFolder"
- Section C.3.7, "customContent"
- Section C.3.8, "attributes"
- Section C.3.9, "attribute"
- Section C.3.10, "parameters"
- Section C.3.11, "schema"
- Section C.3.12, "resource"

For an example of creating a custom catalog definition file, see Section 16.2, "Creating a Custom Resource Catalog."

C.3.1 catalogDefinition

The `catalogDefinition` element defines the overall characteristics of the catalog, such as the location of translated strings.

Example C–2 illustrates the use of `catalogDefinition`.

Example C–2 Example of catalogDefinition

```
<catalogDefinition id="DefaultCatalog" visible="#{true}"
                   resourceBundle="portal.catalog.DefaultCatalogBundle"
                   definitionFilter="portal.catalog.DefaultCatalogFilter"
                   xmlns="http://xmlns.oracle.com/adf/rcs/catalog">
```

Table C–1 describes the keywords of the `catalogDefinition` element. For information about including `attributes` for elements, see Section C.3.8, "attributes."

Table C–1 catalogDefinition Keywords

Keyword	Description
`xmlns`	`http://xmlns.oracle.com/adf/rcs/catalog`
`id`	The identifier used when referring to this catalog definition.
visible	Whether the Resource Catalog must be available for selection at runtime. This attribute takes an EL value and can be used to conditionally display the catalog to specific users or groups. The default value is #{true}, which means that the catalog is displayed at all times.
`resourceBundle`	The name of the resource bundle to be used for obtaining translated attribute values, where `<attribute.... isKey="true"/>`. The `resourceBundle` attribute can also be used at lower levels in the document, such as on individual `attributes` elements. Refer to the XSD for further details.
`definitionFilter`	The fully qualified name of a Java class that implements the `oracle.adf.rc.spi.plugin.catalog.CatalogDefinitionFilter`. You can use a definition filter to dynamically exclude entries from a catalog at runtime. One common use of a definition filter is to provide different views of the same catalog to users with different roles. For example, WebCenter Portal: Spaces uses a definition filter to hide entries for services that the administrator has disabled.

A catalog can have `attributes`, such as title, subject, and description. For information about including `attributes`, see Section C.3.8, "attributes."

C.3.2 folder

The `folder` element defines a catalog folder, used to organize content in the catalog. Folders can contain any number of entries including other folders. You can mix different types of catalog elements in the same folder.

Folders can have `attributes` just like any other entry. For information about including `attributes` for elements, see Section C.3.8, "attributes."

Example C–3 illustrates the use of the `folder` element.

Example C–3 Example of folder

```
<folder visible="#{true}" id="webDevFolder">
  <attributes>
    <attribute value="WEB_DEV_FOLDER.TITLE" attributeId="Title"
               isKey="true"/>
    <attribute value="WEB_DEV_FOLDER.DESCRIPTION" attributeId="Description"
               isKey="true"/>
    <attribute value="WEB_DEV_FOLDER.KEYWORDS" attributeId="Subject"
               isKey="true"/>
    <attribute value="/adf/webcenter/folderlayout_qualifier.png"
```

```
                isKey="false" attributeId="IconURI"/>
  </attributes>
  <contents/>
<folder>
```

Table C–2 describes the keywords of the `folder` element.

Table C–2 folder Keywords

Keyword	Description
`id`	The unique identifier used when referring to this folder.
`visible`	Whether the resource must be rendered in the catalog. This attribute takes an EL value and can be used to conditionally display the resource to specific users or groups. The default value is #{true}, which means that the resource is displayed in the catalog at all times.

Rather than adding the code to the catalog definition source, you may find it easier to add a folder in Design view of the catalog definition file. For more information, see Section 16.2.3.1, "Adding a Folder."

C.3.3 url

The `url` element is used to define the following types of links in a catalog:

- **Task Flow**: Link to a WebCenter Portal service task flow or a custom task flow created in Oracle JDeveloper.
- **Portlet**: Link to any registered portlet producer.
- **Content**: Link to a file or directory from an existing Content Repository connection.
- **Other**: Link to a custom component that you create by providing the XML code for that component.

Example C–4 illustrates the use of the `url` element.

Example C–4 Example of a url Resource

```
<url visible="#{true}"

factoryClass="oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourceFa
ctory"
     id="tagCloud"

url="taskflow://oracle/webcenter/tagging/controller/taskflows/tag-selection.xml#ta
g-selection">
  <attributes>
    <attribute value="TAG_CLOUD.TITLE" attributeId="Title" isKey="true"/>
    <attribute value="TAG_CLOUD.DESCRIPTION" attributeId="Description"
               isKey="true"/>
    <attribute value="TAG_CLOUD.KEYWORDS" attributeId="Subject"
               isKey="true"/>
    <attribute value="oracle.webcenter.tagging"
               attributeId="WEBCENTER_SERVICE_ID" isKey="false"/>
    <attribute value="/adf/webcenter/label_qualifier.png"
               attributeId="IconURI"/>
    <attribute value="height:211px" attributeId="attr.contentStyle"
               isKey="false"/>
```

```
  </attributes>
  <contents/>
</url>
```

Table C–3 describes the keywords for the `url` element.

Table C–3 url Keywords

Keyword	Description
`id`	The unique identifier used when referring to this folder.
visible	Whether the resource must be rendered in the catalog. This attribute takes an EL value and can be used to conditionally display the resource to specific users or groups. The default value is #{true}, which means that the resource is displayed in the catalog at all times.
`factoryClass`	If you are adding a custom component, then specify the factory class of the component. For all other resource types, the `Factory Class` value is populated as soon as you select the type in Design mode. The factory classes for the available resource types are as follows: ■ Task flow: `oracle.webcenter.portalframework.sitestructure.rc.TaskFlowResourceFactory` ■ Portlet: `oracle.webcenter.portalframework.sitestructure.rc.PortletResourceFactory` ■ Content: `oracle.webcenter.content.model.rc.ContentUrlResourceFactory` ■ Data control: `oracle.webcenter.datacomposer.internal.adapter.datacontrol.DataControlContextFactory`
`URL`	The URL to access the resource. Different link types support different URL formats. For more information, see Table 16–1, " URL Format for the Different Resource Types".

Resource links can have attributes, such as title, subject, and description, and parameters. For more information, see Section C.3.8, "attributes" and Section C.3.10, "parameters."

Rather than adding the code to the catalog definition source, you may find it easier to add a link in Design view of the catalog definition file. For more information, see Section 16.2.3.2, "Adding a Link."

C.3.4 includeCatalog

The `includeCatalog` element is used to include a reference to another catalog definition file.

Example C–5 illustrates the use of the `includeCatalog` element.

Example C–5 Example of a Nested Resource Catalog

```
<includeCatalog visible="#{true}" scope="/" id="users-catalog"

catalogId="/oracle/webcenter/portalapp/catalogs/users-catalog.xml">
 <attributes>
```

```
   <attribute value="catalog" isKey="false" attributeId="Title"/>
 </attributes>
</includeCatalog>
```

The `includeCatalog` element supports the same keywords as the `catalogDefinition` element. See Table C–1 for the keywords on the `includeCatalog` element.

A catalog can have `attributes`, such as title, subject, and description. For information about including `attributes` for elements, see Section C.3.8, "attributes."

Rather than adding the code to the catalog definition source, you may find it easier to add a catalog in Design view of the catalog definition file. For more information, see Section 16.2.3.3, "Adding a Resource Catalog."

C.3.5 component

The `component` element is used to include a custom XML resource, such as a simple ADF Faces component or raw HTML, in the catalog. The `component` element must be located within the `contents` section. You can add multiple `component` in a catalog definition file.

The `component` element must contain a `parameters` element with a nested `parameter` element. Set the `id` for the `parameter` element to `xml`. Include the XML fragment inside this `parameter` element. To generate an ADF Faces ID for the component at runtime, set the ID on the component to `#` (hash symbol).

Example C–6 shows the code to add a `Box` component to the Resource Catalog.

Example C–6 Panel Customizable Component in the Catalog Definition File

```
<component id="pc" visible="#{true}"
           factoryClass="oracle.adf.rc.component.XmlComponentFactory">
  <attributes>
    <attribute attributeId="Title" value="BOX" isKey="true"/>
    <attribute attributeId="Description" value="BOX.DESCRIPTION"
               isKey="true"/>
    <attribute attributeId="Subject" value="BOX.KEYWORDS" isKey="true"/>
    <attribute attributeId="IconURI"
               value="/adf/webcenter/panelcustomizable_qualifier.png"/>
  </attributes>
<parameters>
  <parameter id="xml">&lt;cust:panelCustomizable id="#"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/&gt;
  </parameter>
</parameters>
  </parameters>
</component>
```

Table C–4 describes the keywords of the `component` element. For information about including `attributes` for elements, see Section C.3.8, "attributes."

Table C–4 component Keywords

Keyword	Description
`id`	The unique identifier for the component in the catalog definition.

Table C–4 (Cont.) component Keywords

Keyword	Description
visible	Whether the resource must be rendered in the catalog. This attribute takes an EL value and can be used to conditionally display the resource to specific users or groups. The default value is `#{true}`, which means that the resource is displayed in the catalog at all times
`factoryClass`	The name of a Java class that implements the `oracle.adf.rc.component.ComponentFactory` interface and generates a Resource Catalog item based on a set of parameters. Each `ComponentFactory` implementation defines the set of parameters it requires. For example, `XmlComponentFactory` requires a single parameter named `xml` and the value of the parameter is a block of XML representing a component or components that can be inserted into a page.

A custom component can have attributes, such as title, subject, and description, and parameters. For more information, see Section C.3.8, "attributes" and Section C.3.10, "parameters."

You may find it easier to add a custom component in Design view of the catalog definition file. For more information, see Section 16.2.3.4, "Adding a Component."

C.3.6 customFolder

The `customFolder` element defines content generated using Resource Catalog adapters. This element renders a folder that is populated with content dynamically at runtime. The catalog displays the folder even if there are no resources inside it.

Example C–7 illustrates the use of the `customFolder` element to render task flows from the Analytics service.

Example C–7 Example of a customFolder Resource

```
<customFolder
factoryClass="oracle.webcenter.analytics.view.rc.ASServiceContextFactory"
id="analyticsServiceFolder" visible="#{true}">
         <attributes
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle">
            <attribute attributeId="Description" isKey="true"
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"
value="ANALYTICS_FOLDER.DESCRIPTION"/>
            <attribute attributeId="IconURI" isKey="false"
value="/adf/webcenter/folderanalytics_qualifier.png"/>
            <attribute attributeId="Subject" isKey="true"
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"
value="ANALYTICS_FOLDER.KEYWORDS"/>
            <attribute attributeId="WEBCENTER_SERVICE_ID" isKey="false"
value="oracle.webcenter.analytics"/>
            <attribute attributeId="attr.text" isKey="false" value="#{uib_o_w_s_r_
DefaultGroupSpaceCatalog['ANALYTICS_FOLDER.TITLE']}"/>
            <attribute attributeId="Title" isKey="true"
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"
value="ANALYTICS_FOLDER.TITLE"/>
         </attributes>
         <parameters/>
      </customFolder>
```

Table C–5 describes the keywords for the `customFolder` element.

Table C–5 customFolder Keywords

Keyword	Description
`id`	The unique identifier used when referring to this folder.
visible	Whether the resource must be rendered in the catalog. This attribute takes an EL value and can be used to conditionally display the resource to specific users or groups. The default value is #{true}, which means that the resource is displayed in the catalog at all times.
factoryClass	The factory class of the service. See Section C.4, "Factory Classes Available for Adding Dynamic Resources to the Catalog" for information about the supported factory classes.

A custom folder can have attributes, such as title, subject, and description, and parameters. For more information, see Section C.3.8, "attributes" and Section C.3.10, "parameters."

Rather than adding the code to the catalog definition source, you may find it easier to add a custom folder in Design view of the catalog definition file. For more information, see Section 16.2.3.5, "Adding a Custom Folder."

C.3.7 customContent

The `customContent` element defines content generated using Resource Catalog adapters. This element renders a folder that contains resources that are generated dynamically at runtime. Unlike the `customFolder` element, the `customContent` element renders a folder in the catalog only if the folder contains resources.

Example C–8 illustrates the use of the `customContent` element.

Example C–8 Example of a customContent Resource

```
<customContent
contentProviderClass="com.oracle.ensemble.interop.adf.EnsembleContentProvider"
               id="ensembleContentProvider" visible="true">
  <attributes
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"/>
</customContent>
```

Table C–3 describes the keywords for the `customContent` element.

Table C–6 customContent Keywords

Keyword	Description
`id`	The unique identifier used when referring to this folder.
visible	Whether the resource must be rendered in the catalog. This attribute takes an EL value and can be used to conditionally display the resource to specific users or groups. The default value is #{true}, which means that the resource is displayed in the catalog at all times.
contentProviderClass	The fully qualified name of a class that implements the interface `oracle.adf.rc.spi.plugin.catalog.CustomContentProviderV2`.

A custom content provider can have attributes, such as title, subject, and description, and parameters. For more information, see Section C.3.8, "attributes" and Section C.3.10, "parameters."

Rather than adding the code to the catalog definition source, you may find it easier to add a custom content folder in Design view of the catalog definition file. For more information, see Section 16.2.3.6, "Adding a Custom Content Provider."

C.3.8 attributes

Entries in a Resource Catalog have associated attributes that describe the entry. Values for some attributes can be obtained directly from the resources or from the `name` and `description` attributes on each XML element in the catalog definition. However, these values may not always be appropriate for displaying to end users, and often they are not available in different languages. Attributes defined using the `attributes` element in the catalog definition let you override those available from other sources. In addition, these attributes can be defined in terms of a key to a resource bundle so the actual values displayed reflect the current user's locale.

Within the `attributes` element you can insert any number of individual `attribute` elements that define the attributes you want to apply to a Resource Catalog entry.

Example C–9 illustrates the use of the `attributes` element.

Example C–9 Example of attributes and attribute Elements

```
<attributes>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.TITLE" attributeId="Title"
  isKey="true"/>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.DESCRIPTION" attributeId="Description"
  isKey="true"/>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.KEYWORDS" attributeId="Subject"
  isKey="true"/>
 <attribute value="oracle.webcenter.collab.announcement"
  attributeId="WEBCENTER_SERVICE_ID" isKey="false"/>
</attributes>
```

Table C–7 describes the keywords of the `attributes` element.

Table C–7 attributes Keywords

Keyword	Description
`resourceBundle`	The name of the resource bundle to be used for obtaining translated attribute values.

C.3.9 attribute

The `attribute` element defines an attribute for a Resource Catalog entry that corresponds to a descriptor in the `schema` element. For information about attributes, see Section C.3.8, "attributes." Composer uses the following attributes:

- Title
- Description
- Subject
- IconURI
- ToolTip

You should include at least a `Title` and `Description` attribute for each catalog entry. A `Subject` attribute is also highly recommended to facilitate keyword searching of the Resource Catalog. The `Title` attribute overrides the `name` attribute on the parent element and is used to declare a display name that is translated through a resource bundle. Similarly, the `Description` attribute overrides the `description` attribute on the parent element and is used to declare a description that is translated through a resource bundle.

You can declare your own attributes by including them in the `schema` element. For more information, see Section C.3.11, "schema."

Example C–10 illustrates the use of the `attribute` element.

Example C–10 Example of attributes and attribute Elements

```
<attributes>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.TITLE" attributeId="Title"
  isKey="true"/>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.DESCRIPTION" attributeId="Description"
  isKey="true"/>
 <attribute value="ANNOUNCEMENTS_MAIN_VIEW.KEYWORDS" attributeId="Subject"
  isKey="true"/>
 <attribute value="oracle.webcenter.collab.announcement"
  attributeId="WEBCENTER_SERVICE_ID" isKey="false"/>
</attributes>
```

Table C–8 describes the keywords of the `attribute` element.

Table C–8 attribute Keywords

Keyword	Description
`attributeId`	The identifier of the attribute as declared in the `schema` element.
`value`	The value of the attribute.
`isKey`	A flag indicating whether the value is a literal value (`false`) or a key to a resource bundle (`true`). In most cases, attributes should be resource bundle keys and the values translated through a resource bundle. The resource bundle can be declared at the catalog level or on various lower level elements such as `attributes`.
`resourceBundle`	The name of the resource bundle to be used for obtaining translated attribute values.

For task flows included in your Resource Catalog, you can use the `attribute` element to pass task flow parameters and attributes of the enclosing `Show Detail Frame` component. You can do this by prefixing the `attributeId` values for the task flow parameters and `Show Detail Frame` attributes with **`parameter.`** and **`attr.`** respectively as shown in the following example:

```
<attributes>
  <attribute value="dark" attributeId="attr.background"
   isKey="false"/>
  <attribute value="#{myBean.myParam1}" attributeId="parameter.myParam1"
   isKey="false"/>
</attributes>
```

For more information, see Section 16.2.9, "How to Define Task Flow Parameters and Attributes of the Enclosing Show Detail Frames for Task Flows."

C.3.10 parameters

A few of the Resource Catalog components, such as links, custom components, and custom folders, support parameters. Parameters are used to provide the XML code for a custom component or bind the resource to other artifacts in the application. You can use the `parameters` element to define parameters on a resource. The `parameters` element must be included inside the tag for the resource. Each `parameters` element can have multiple `parameter` tags. You must provide an ID and a value for each parameter that you define.

When you define a parameter for a custom XML component, you must specify the parameter ID to be `xml` and the value is the code for the custom component. In the code, the component's ID must be `#` so that it is generated at runtime. In addition, you must specify the namespace for the component.

Example C–11 illustrates the use of the `parameters` element in a custom XML component definition.

Example C–11 Example of parameters and parameter Elements

```
<parameters>
  <parameter id="xml">&lt;cust:panelCustomizable id="#"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/&gt;
  </parameter>
</parameters>
```

C.3.11 schema

The `schema` element contains one or more `descriptor` elements. The `descriptor` elements provide the Resource Catalog service with information about the attributes used within the catalog, such as whether they are searchable. Out-of-the-box, the attributes Title, Description, Subject, ToolTip, and IconURI are supported for a resource. You can add other custom attributes that you want to expose on the resource.

Example C–12 illustrates the use of the `schema` element.

Example C–12 Example of schema

```
<schema resourceBundle="oracle.adf.rc.attribute.nls.AttributeBundle"
          xmlns="http://xmlns.oracle.com/adf/rcs/catalog">
    <descriptor labelKey="TITLE.PROMPT_KEY" endUserVisible="true"
                multivalue="false" searchable="true" attributeId="Title"
                shortLabelKey="TITLE.SHORT_PROMPT_KEY"
                xmlns="http://xmlns.oracle.com/adf/rcs/catalog"/>
    <descriptor labelKey="DESCRIPTION.PROMPT_KEY" endUserVisible="true"
                multivalue="false" searchable="true" attributeId="Description"
                shortLabelKey="DESCRIPTION.SHORT_PROMPT_KEY"
                xmlns="http://xmlns.oracle.com/adf/rcs/catalog"/>
    <descriptor labelKey="SUBJECT.PROMPT_KEY" endUserVisible="true"
                multivalue="false" searchable="true" attributeId="Subject"
                shortLabelKey="SUBJECT.SHORT_PROMPT_KEY"
                xmlns="http://xmlns.oracle.com/adf/rcs/catalog"/>
    <descriptor labelKey="TOOL_TIP.PROMPT_KEY" endUserVisible="true"
                multivalue="false" searchable="false" attributeId="ToolTip"
                shortLabelKey="TOOL_TIP.SHORT_PROMPT_KEY"
                xmlns="http://xmlns.oracle.com/adf/rcs/catalog"/>
    <descriptor labelKey="ICON_URI.PROMPT_KEY" endUserVisible="false"
                multivalue="false" searchable="false" attributeId="IconURI"
                shortLabelKey="ICON_URI.SHORT_PROMPT_KEY"
```

```
                xmlns="http://xmlns.oracle.com/adf/rcs/catalog"/>
  </schema>
```

Table C–9 describes the keywords of the `descriptor` element.

Table C–9 descriptor Keywords

Keyword	Description
`attributeId`	The identifier of the attribute being described.
`searchable`	Indicates whether the attribute is included in Resource Catalog searches (`true`) or not (`false`).
`multivalue`	Indicates whether the attribute can have multiple values (`true`) or not (`false`).
`endUserVisible`	Indicates whether the attribute should be exposed to end users (`true`) or not (`false`).
`resourceBundle`	The name of the resource bundle used to translate the label keys.
`labelKey`	The resource bundle key for the attribute's label.
`shortLabelKey`	The resource bundle key for the attribute's short label.

C.3.12 resource

The `resource` element is used to declare an individual resource, such as a task flow, in the catalog. You would need to use this element if you are using a standalone version of Composer (that is, Composer without WebCenter Portal: Framework).

> **Note:** The `url` element, discussed in Section C.3.3, "url," also enables you to declare a resource in the catalog. However, `url` is available only if you have installed WebCenter Portal: Framework and have the associated shared libraries.

Example C–13 illustrates the use of the `resource` element.

Example C–13 Example of resource

```
<resource id="weatherTF"
      repository="application.classpath"
      path="TaskFlow.jar/ADF_
TaskFlow/oracle+composer+internal+test+weather-tfdefn.xml#weather_tf_defn"
      visible="#{true}">
  <attributes>
    <attribute value="/images/bulbgauge_ena.png" attributeId="IconURI"/>
    <attribute value="never" attributeId="attr.showResizer"/>
    <attribute value="Weather in Bangalore is rainy"
               attributeId="parameter.message"/>
    <attribute value="WeatherBee" isKey="false" attributeId="Title"/>
    <attribute value="Displays the weather of your city" isKey="false"
               attributeId="Description"/>
    <attribute value="Weather widget" isKey="false" attributeId="ToolTip"/>
  </attributes>
  <parameters/>
</resource>
```

Table C–10 describes the keywords of the `resource` element. For information about including `attributes` for elements, see Section C.3.8, "attributes."

Table C–10 resource Kaywords

Keyword	Description
`id`	The identifier for the element in the catalog. Identifiers are required and must be unique at the folder level.
`repository`	The path to access ADF libraries included in your classpath.

`path`	The path to the resource. For a task flow in an ADF Library, the path is of the following form:
	`path_to_jar/ADF_TaskFlow/task_flow_path`
	To obtain the `task_flow_path`, create a file system connection in Application Connections and navigate to the task flow in your ADF Library. The tooltip for the task flow shows the fully qualified ID of the task flow. Take this value and replace all occurrences of "/" with "+".
	For example:
	If the tooltip is:
	`oracle/webcenter/collab/announcement/view/taskflows/main-view-definition.xml#announcement-main-view`
	then `task_flow_path` would be:
	`oracle+webcenter+collab+announcement+view+taskflows+main-view-definition.xml#announcement-main-view`

C.4 Factory Classes Available for Adding Dynamic Resources to the Catalog

Resource Catalog adapters are used to add resources to a catalog dynamically, for example, a folder to display all available content repository connections to users. As the content of such a folder cannot be determined when you create the catalog, the adapter ensures that the folder is populated dynamically.

To use an adapter, you must specify the factory class as part of the resource definition while adding the resource. This section describes the supported factory classes. It includes the following sections:

- Section C.4.1, "XMLComponentFactory"
- Section C.4.2, "ContentUrlResourceFactory"
- Section C.4.3, "CustomFolderContextFactory"
- Section C.4.4, "DefaultFolderContextFactory"
- Section C.4.5, "ContentListFactory"
- Section C.4.6, "EnsembleContentProvider"

C.4.1 XMLComponentFactory

Use the `oracle.adf.rc.component.XmlComponentFactory` class inside a `<component>` construct to include an XML component in the catalog. The `XmlComponentFactory` class requires a single parameter named `xml` and the value of the parameter is a block of XML representing one or more components that can be added to a page. The following example shows the code to add a `Panel Customizable` component to the Resource Catalog:

```
<component id="pc" factoryClass="oracle.adf.rc.component.XmlComponentFactory">
```

```
  <attributes>
    <attribute attributeId="Title" value="Box"/>
    <attribute attributeId="Description" value="Container for other objects"/>
    <attribute attributeId="IconURI" value="/adf/webcenter/panelcustomizable_
qualifier.png"/>
  </attributes>
  <parameters>
    <parameter id="xml">
      <cust:panelCustomizable id="#"
xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable"/>
    </parameter>
  </parameters>
</component>
```

For the detailed steps, see Section 16.2.3.4, "Adding a Component."

C.4.2 ContentUrlResourceFactory

The `ContentUrlResourceFactory` class is used to include portal framework content from UCM connections. It is used when you add a URL resource of type Content to the catalog. The following example shows how the `ContentUrlResourceFactory` class can be used:

```
<url id="contentItem"
     factoryClass="oracle.webcenter.content.model.rc.ContentUrlResourceFactory"
     url="content://UCM/PersonalSpaces/weblogic/content.xml">
<attributes>
  <attribute value="Personal Document" isKey="false" attributeId="Title"/>
</attributes>
<parameters>
  <parameter id="templateView">SpaceTemplate</parameter>
  </parameters>
</url>
```

where `id` is the unique identifier for the resource, and `url` is the URL to access the resource. The URL must begin with `content:` if you use `ContentUrlResourceFactory`. In addition, you can add a `templateView` parameter to specify the Content Presenter display template to be used.

For more information, see Section 16.2.3.2, "Adding a Link."

C.4.3 CustomFolderContextFactory

The `ContentFolderContextFactory` class is used to list all the Content Repository connections available to the application. You can add a `customFolder` resource to the catalog and specify this factory class, as shown in the following example:

```
<customFolder id="AllConnections"

factoryClass="oracle.webcenter.content.model.rc.CustomFolderContextFactory" />
```

For more information about custom folders, see Section 16.2.3.5, "Adding a Custom Folder."

C.4.4 DefaultFolderContextFactory

In the WebCenter Portal: Spaces context, the `DefaultFolderContextFactory` class is used to render the folder for the current space. If the user is in the Home space, then

the user's personal folder is rendered. You can add a `customFolder` resource to the catalog and specify this factory class, as shown in the following example:

```
<customFolder id="GroupDocuments"

factoryClass="oracle.webcenter.doclib.model.rc.DefaultFolderContextFactory" />
```

For more information about custom folders, see Section 16.2.3.5, "Adding a Custom Folder."

C.4.5 ContentListFactory

The `ContentListFactory` class is used to add a dynamic list of folders to the catalog. You can add a `customFolder` resource to the catalog and specify this factory class, as shown in the following example:

```
<customFolder id="customFolder-folder" visible="#{true}" path=""
              factoryClass="oracle.webcenter.content.model.rc.ContentListFactory">
  <attributes>
    <attribute attributeId="Title" value="Folders List" isKey="false"/>
  </attributes>
  <parameters>
    <parameter id="datasource">UCM#dCollectionID:60</parameter>
    <parameter id="datasourceType">dsTypeFolderContents</parameter>
  </parameters>
 </customFolder>
```

The `ContentListFactory` class supports the following parameters:

- `datasource`: The data source of the content. The value depends on the value of `datasourceType`.
- `datasourceType`: The data source type of the content. Valid values are `dsTypeSingleNode`, `dsTypeFolderContents`, `dsTypeQueryExpression`, `dsTypeMultiNode`, and `dsTypeScenarioResults`.
- `templateView`: To specify the Content Presenter display template to be used.

For more information about using Content Presenter, see Chapter 31, "Adding Content Task Flows and Document Components to a Portal Page."

C.4.6 EnsembleContentProvider

The `EnsembleContentProvider` class is used to display a dynamic list of all registered pagelet producers. You can add a `customContent` resource to the catalog and specify this factory class, as shown in the following example:

```
<customContent
contentProviderClass="com.oracle.ensemble.interop.adf.EnsembleContentProvider"
               id="ensembleContentProvider" visible="true">
  <attributes
resourceBundle="oracle.webcenter.spaces.resource.DefaultGroupSpaceCatalogBundle"/>
</customContent>
```

For more information about custom content providers, see Section 16.2.3.6, "Adding a Custom Content Provider."

D

Guidelines for Creating Task Flows to Be Used in Composer-Enabled Pages

Task flows are reusable building blocks in Oracle ADF. Oracle WebCenter Portal provides task flows for you, or you can build your own task flows and add them to your Framework application pages. When you add task flows to customizable pages (that is, Composer-enabled pages), users can personalize, customize, and edit these task flows at runtime. It is imperative that you design your task flows according to the rules set forth in Oracle WebCenter Portal publications. Failure to do so can result in problems when the task flows are consumed in portal pages, such as too much white space, too many scroll bars, or slow, jerky, and unpredictable responses when displaying large data sets.

To prevent these issues and provide a consistent user experience in terms of look and feel, layout design, and interaction, follow the instructions in this document for:

- Creating highly performant, cost-effective task flows
- Consuming these task flows effectively in your application by implementing task flow parameters and customization and personalization options

This appendix contains the following topics:

- Section D.1, "Guidelines for Effective Geometry Management and Pagination"
- Section D.2, "Guidelines for Efficient Use of Task Flow Parameters and Customization and Personalization Options"
- Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines"
- Section D.4, "Conclusion"

D.1 Guidelines for Effective Geometry Management and Pagination

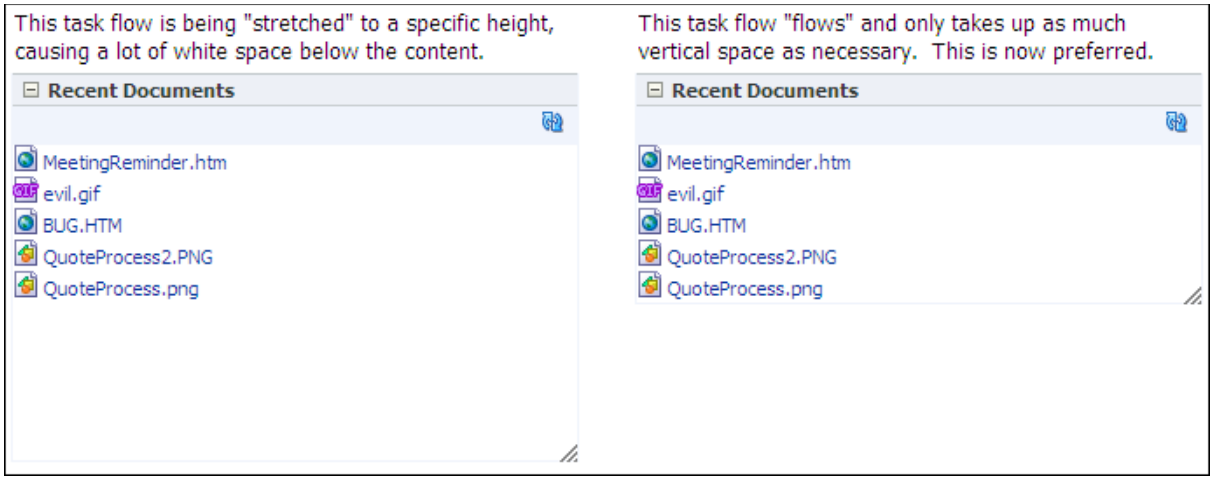
Effective geometry management results in efficient sizing and placement of task flows on the page. If the task flows are not designed properly, you end up with pages that have a slow response time or lots of unnecessary scroll bars or white spaces between components. For example, if you include a task flow containing very little data inside a container that stretches it (for example, a `Show Detail Frame` component), there is a lot of white space below the data in the task flow. Poor design can also cause scrolling to be slow and unpredictable in a task flow with a table displaying a large amount of data. To avoid such issues, be sure to design your task flows so that they flow and use a conventional pagination model when displaying large data sets.

Follow the guidelines in this section when designing new task flows for Framework applications. Also, be sure to adhere to these guidelines when you edit existing Framework application task flows.

D.1.1 Guideline 1: Create Task Flows that Flow

To avoid unnecessary white spaces in your task flows, you must ensure that the task flow content flows and is not stretched by the task flow container. A *stretching* task flow is one that is stretched to the height of the container, whereas a *flowing* task flow is one whose height is determined by its content, rather than by a stretching container or by itself having a fixed height. A flowing task flow with a small amount of data is shorter than one with large amounts of data, as shown in Figure D–1.

Figure D–1 Difference Between a Task Flow that Stretches and One that Flows



To design a flowing task flow, you must ensure that:

- The child component of the task flow does not have a fixed height.
- The `Show Detail Frame` component surrounding the task flow region does not have a fixed height (specified using its `contentStyle` attribute).

The following two examples illustrate how you might include flowing components inside task flows.

D.1.1.1 Example 1: Oracle ADF Faces Table Component with Its autoHeightRows Attribute Set to a Specific Value

Add an Oracle ADF Faces `Table` component within the task flow and set the `autoHeightRows` attribute on the `Table` to a specific value. By defining this attribute, the table height is adjusted according to the content (flow behavior), rather than the table having a fixed height or being stretched into a container (stretch behavior). The `autoHeightRows` attribute setting results in table data being displayed as follows:

- If you provide a value `H` for `autoHeightRows`, the table grows to a maximum of `H` rows.
- If the number of records returned from the data source is less than `H`, the table height wraps to that number.
- If the number of records is more than `H`, the table displays `H` rows and provides a scrollbar to view the remaining rows.

- If the `autoHeightRows` attribute is not used, the table displays at a fixed default height irrespective of the amount of data displayed.

Instead of specifying a fixed value for the `autoHeightRows` attribute, you can also provide an EL value so that the attribute is populated based on some logic you specify. You can enable customization and personalization on the task flow and provide an option for users to specify a value for this attribute. A value specified using the personalization option overrides a value specified using the customization option.

Example D–1 shows the attributes of a sample ADF Faces `table` component that is included inside a task flow called `recent-pages-task-flow-definition`. The `autoHeightRows` attribute is set to an EL expression that references a Java bean, `recentPagesBean`. This bean contains the logic to populate this attribute with a value provided by the end user (either as a personalization or a customization).

Example D–1 The autoHeightRows Attribute on a Sample ADF Faces Table

```
<af:table value="#{pageFlowScope.recentPagesBean.pages}"
          var="row" rowBandingInterval="0" id="t1"
          rowSelection="none" columnBandingInterval="0"
          width="100%" inlineStyle="border:none;width:100%;"
          columnStretching="last" horizontalGridVisible="true"
          verticalGridVisible="false"
          autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_show_
at_a_time}"
            contentDelivery="immediate"
           fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}">
      .
      .
      .
</af:table>
```

For a detailed description of the sample task flow displaying personalization and customization options for specifying the `autoHeightRows` value, see Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines."

To learn more about the `autoHeightRows` attribute, including requirements such as setting the `contentDelivery` attribute to `immediate`, see the tag documentation for `<af:table>` in the *Oracle Fusion Middleware Tag Reference for Oracle ADF Faces*.

D.1.1.2 Example 2: Oracle ADF Faces Iterator Component Inside a Panel Group Layout with a Scrolling Layout

Add a `Panel Group Layout` component within the task flow and set its `layout` attribute to `scroll`. Then add an `Iterator` inside the `Panel Group Layout`. Add your content inside the `Iterator`. The content from the `Iterator` flows inside the `Panel Group Layout`.

Since the height of the `Panel Group Layout` depends on the content from the `Iterator`, specify a minimum and maximum height for the component by using `min-height` and `max-height` in the `Panel Group Layout`'s `inlineStyle` attribute.

D.1.2 Guideline 2: Limit the Number of Records Displayed in a Task Flow

To avoid a slow and jerky response when scrolling large data sets in your task flow, you must limit the number of records to be fetched from the data source at a time. For example, if you have included an ADF faces `Table` component inside your task flow,

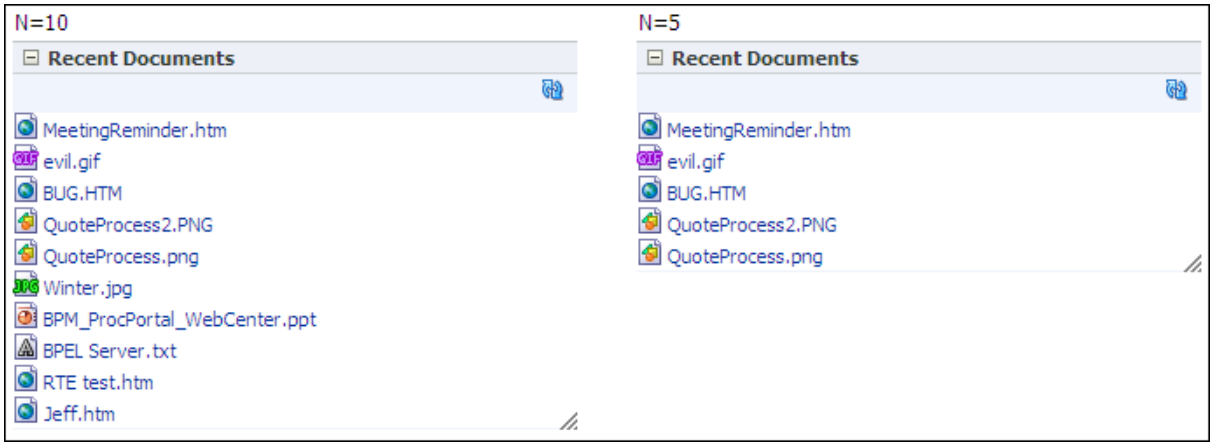
limit the number of records to be displayed in the table. Use the `fetchSize` attribute on the `Table` to specify the number of records to be fetched from the data source in a single query. If you specify a fetch size of `N`, only `N` records are sent to the client from the data source.

The `fetchSize` and `autoHeightRows` attributes together control the number of rows displayed in the table. The relationship between these attributes is as follows:

- If `fetchSize` equals `autoHeightRows`, all records are displayed in the table.
- If `fetchSize` is less than `autoHeightRows`, the table is cropped at the `fetchSize` value.
- If `fetchSize` is greater than `autoHeightRows`, you can decide whether to display only as many records as the `autoHeightRows` value or provide pagination controls to display the remaining records. For example, you might include a More link at the bottom of the task flow or include pagination using Previous and Next links. For more information, see Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines."

Figure D–2 shows the difference between two task flows containing tables with equal `autoHeightRows` and `fetchSize` values of 10 and 5. The task flows wrap to the number of records displayed.

Figure D–2 Difference Between Task Flows with Different Fetch Size Values



Instead of specifying a fixed value for `fetchSize`, you can provide an EL value so that the attribute is populated based on some logic you specify. You can enable customization and personalization on the task flow and provide an option for users to specify a value for this attribute. A value specified using the personalization option overrides a value specified using the customization option.

The following example shows the attributes of a sample ADF Faces `Table` component that is included inside a task flow called `recent-pages-task-flow-definition`. The `fetchSize` attribute is set to an EL expression that references a Java bean, `recentPagesBean`. This bean contains the logic to populate this attribute with a customization value provided by the end user.

```
<af:table value="#{pageFlowScope.recentPagesBean.pages}"
          var="row" rowBandingInterval="0" id="t1"
          rowSelection="none" columnBandingInterval="0"
          width="100%" inlineStyle="border:none;width:100%;"
          columnStretching="last" horizontalGridVisible="true"
          verticalGridVisible="false"
```

```
          autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_
show_at_a_time}"
          contentDelivery="immediate"
          fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}">
      .
      .
      .
</af:table>
```

For a detailed description of the sample task flow displaying a customization option for specifying the `fetchSize` value, see Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines."

To learn more about the `fetchSize` attribute, see the tag documentation for `<af:table>` in the *Oracle Fusion Middleware Tag Reference for Oracle ADF Faces*.

D.1.3 Guideline 3: Specify a Minimum Height for the Task Flow

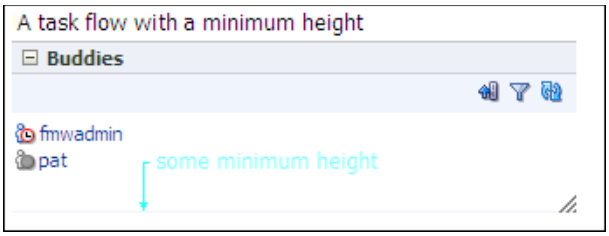
You can ensure that the task flow wraps to the height of the content by adhering to the first guideline, described in Section D.1.1, "Guideline 1: Create Task Flows that Flow."

To prevent the task flow from collapsing completely when there are no records, or seeming too short when there are only one or two records, specify a minimum height for the content.

For example, if your task flow contains a table with an `autoHeightRows` value specified, you can specify a `min-height` value using the table's `inlineStyle` attribute.

Figure D–3 shows a task flow displayed with a reasonable height even though it has only two rows.

Figure D–3 Task Flow with a Minimum Height Specified



D.1.4 Guideline 4: Include UI for Accessing Data Beyond the Display Limit

If a task flow is showing ten records, but the data source contains 100 records, you can design the task flow to do one of the following depending on the nature of data in the task flow:

- Display only ten records.

  For example, a Popular Discussion Topics task flow may use this first option if it is considered sufficient to show the ten most popular topics and not necessary to let users see even the eleventh most popular topic.

- Show a More link at the bottom of the task flow that opens a view with the ability to show more records.

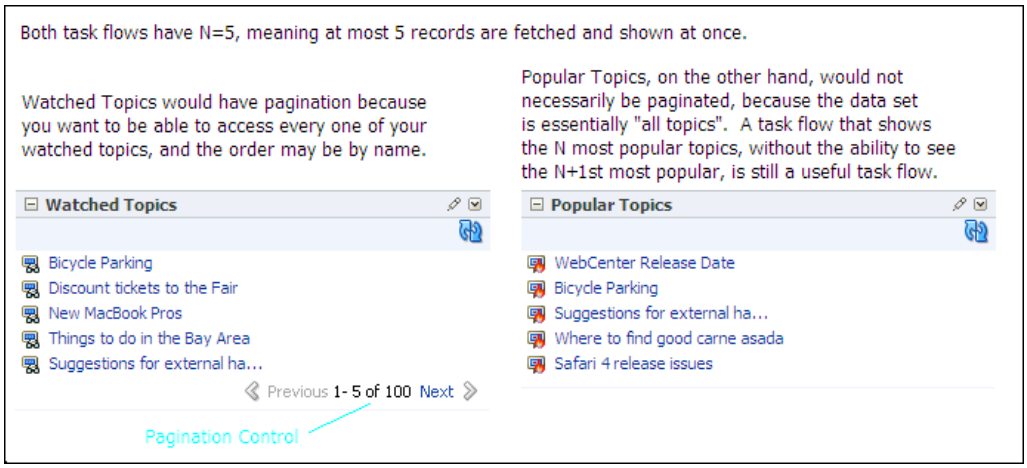
  For example, the People Connections task flow on a Spaces page displays a More link at the bottom.

- Show a pagination control with Previous and Next links.

  For example, a Watched Discussion Topics task flow may display pagination controls, as users typically want to see all of their watched topics. Pagination is a common choice for task flows in which the data is not inherently ranked in order of importance or can be sorted in different ways, and where it is important to be able to access any record in the data set. The pagination control can be placed either above or below the content, depending on the nature of the content.

Figure D–4 shows the Watched Topics and Popular Topics task flows containing tables having a `fetchSize` of 5 but both having more than five records in the data source. The Watched Topics task flow displays five records with pagination at the bottom, but the Popular Topics task flow displays only five records with no UI option to view other records.

Figure D–4 Task Flows With and Without Pagination Controls



Pagination Specifics

Here are a few simple recommendations for creating a pagination control:

- Add a pagination message in the format `1 to N of M [prev][next].`
- Align the pagination elements to the right so that the Previous and Next links do not move as you navigate pages.
- Ensure that the Next or Previous option is grayed out when it is not required.

For an example of how to use pagination controls, see Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines."

You can customize pagination controls using the following parameters:

- Position of Previous and Next links
- Labels for Previous and Next links, for example, Newer and Older
- Icons used with Previous and Next links
- The words or symbols used to represent to and of, for example, `1-5 of 100`
- Options for permitting random access, for example, `Prev 1 2 3 ... Next Last`

D.2 Guidelines for Efficient Use of Task Flow Parameters and Customization and Personalization Options

While creating a task flow, you can create parameters and provide options for users to customize and personalize the task flow. You can enable privileged users to customize the task flow for all users (customization) and edit the task flow's properties and parameters in Composer. You can enable all users viewing the page to customize their view of the page (personalization). This section provides guidelines for implementing task flow parameters and the customization and personalization options.

Table D–1 provides an overview of the various task flow editing options available to users.

Table D–1 Options Used to Define Task Flow Appearance and Behavior

Option	Description	When to Use	Who can See It
Parameters	A small set of ideally scalar values, used to program a task flow or set its context.Each parameter must have a good display name and description (which must include possible values and examples).	Use for a context like the city or pin code so that you can add multiple instances of the task flow, each showing different information or programmed to show information based on receiving a value from the page, programmatically or using the URL. In Spaces, you can use a parameter for a context like space name. Precedence: 3 This value overrides built-in values in the task flow.	Power users (or users with Edit or Customize permission) can see parameters in Composer's Component Properties dialog while editing a page. In Spaces, moderators can see and edit task flow parameters. When a WebCenter Portal service task flow is exposed as a portlet in Oracle Portal, page designers can see these parameters while setting up the portlet.

Table D–1 (Cont.) Options Used to Define Task Flow Appearance and Behavior

Option	Description	When to Use	Who can See It
Customization/Preferences	These two are synonymous.Customizations are typically considered changes a business user makes to the task flow. These changes are available to all viewers of the page and are used to specify values for options defined in the task flow. For example, users can decide whether to display the author name or revision date for a document. Preferences are referred to as a definition of the content in the task. For example, a query definition in the Document Viewer task flow, or a query in OmniPortlet.	Use these when the task flow must provide a user interface that assists the user in defining the task flow behavior. That is, when Lists of Values or specific UI components such as images or radio buttons guide the user better, or when the task flow setup is a multiple step wizard-style interface. Such assistance cannot be provided through parameters because they are strictly scalar values, with no assistance other than a description. Precedence: 2 If there is an identical parameter value (for example, space name), customization overrides that value.	Business users (or users with Edit or Customize permission) editing a page in Composer can see the customization or preferences options. Typically, customization or preferences options are available on the chrome surrounding the task flow in Edit mode. However, users with Edit or Customize permission can also see these options on the chrome in View mode of the page. When a WebCenter Portal task flow is exposed as a portlet in Oracle Portal, the page designer sees these as Edit Defaults options while setting up the portlet.
Personalization	A subset of the customization and preferences options. You cannot have personalization options that are not also customization options.	Use these to allow all viewers of the page to tweak and tune the visual aspects of the task flow or the amount of data shown to them. For example, to set the number of items to display. Precedence: 1 If there is an identical customization value (for example, number of items), personalization overrides that value.	All end users with View permission can see these options. Personalization options are available on the chrome surrounding the task flow. When WebCenter Portal task flows are exposed as portlets in Oracle Portal, the Personalization options allow end users to tweak the same visual aspects.
Implicit Personalization	Options that are persisted for end users from session to session. Implicit personalization options are not available on the chrome surrounding the task flow, but are actions that can be performed directly in the body of the task flow. For example, sorting and column resizing are implicit personalization options.	Use these options sparingly and only for options that are frequently changing. Options like column resizing and sorting are available for free from Oracle ADF.	All end users with View permission on the page can perform personalizations.

To improve application performance, follow the guidelines in this section while implementing parameters and customization and personalization options in task flows.

D.2.1 Guidelines for Implementing Task Flow Parameters

At design time (in JDeveloper), you can create parameters by defining them in the task flow definition XML file. At runtime, users with Edit or Customize permission can edit task flow parameters in the Component Properties dialog in Composer. To provide a good user experience, you must adhere to the following guidelines when creating parameters:

- Provide a display name that is meaningful.
- Provide a description for each parameter, along with examples and the possible values it can take.

The following example shows the code to define a parameter called `layoutStyle` in the task flow definition file:

```
<input-parameter-definition>
  <name>layoutStyle</name>
  <description> Used to control whether the documents must display as a list, with
icons, or with document details.
Allowed values are list, icon, and detail.</description>
  <display-name>Layout Style</display-name>
</input-parameter-definition>
```

D.2.2 Guidelines for Implementing Customization

Users with Edit or Customize permission can access customization options from the chrome surrounding the task flow.

You must adhere to the following guidelines while implementing customization in your task flow:

- Customization options must be a subset of the task flow's parameters.
- Implement a view fragment within the task flow and expose an outcome to open this view in a separate dialog. For example, the control flow rule to invoke a view called `advancedEditPopupView` in a dialog is as follows:

  ```
  <control-flow-case>
    <from-outcome>dialog:advancedEditPopup</from-outcome>
    <to-activity-id>advancedEditPopupView</to-activity-id>
  </control-flow-case>
  ```

  Adhering to this guideline is mandatory for new task flows.

- Implement a view fragment within the task flow and expose an outcome to transition inline to this view. For example, the control flow rule to invoke a view called `advancedEditView` within the same screen is as follows:

  ```
  <control-flow-case>
    <from-outcome>advancedEdit</from-outcome>
    <to-activity-id>advancedEditView</to-activity-id>
  </control-flow-case>
  ```

  Although you can implement a task flow in this way, it is recommended that you expose an outcome to open the view in a dialog.

- Ensure that customizations are saved in the MDS level set up by the application. Unlike task flow parameters, which are stored in the task flow definition itself, customizations are saved in the Metadata Storage (MDS). If you do not explicitly specify the MDS layer to be used to save task flow customizations, customization metadata and the consuming application metadata may get stored in different layers.

To ensure that customizations are saved to the same layer as application metadata, you must clone the application's MDS session in the task flow's preference file and reference that while configuring customization. For an example of how this is implemented, see Section D.3, "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines."

D.2.3 Guidelines for Implementing Personalization

All users viewing the page can access personalization options from the chrome surrounding the task flow in View mode of the page.

You must adhere to the following guidelines while implementing personalization in your task flow:

- Ensure that personalization options are a subset of the customization options you define on the task flow.
- If a key aspect of the task flow is to show objects—for example, documents, users, or requests—then you must include a personalization option for users to set the maximum number of objects to display.
- Implement a view fragment within the task flow, and expose an outcome to open this view in a dialog. For example, the control flow rule to invoke a view called `simpleEditPopupView` in a dialog is as follows:

  ```
  <control-flow-case>
    <from-outcome>dialog:simpleEditPopup</from-outcome>
    <to-activity-id>simpleEditPopupView</to-activity-id>
  </control-flow-case>
  ```

  Adhering to this guideline is mandatory for new task flows.
- Implement a view fragment within the task flow and expose an outcome to transition inline to the view. For example, the control flow rule to invoke a view called `simpleEditView` within the same screen is as follows:

  ```
  <control-flow-case>
    <from-outcome>simpleEdit</from-outcome>
    <to-activity-id>simpleEditView</to-activity-id>
  </control-flow-case>
  ```

  Although you can implement a task flow in this way, it is recommended that you make a transition to bring this view up as a dialog.
- Ensure that personalizations are written to the MDS level set up by the application. Unlike task flow parameters, which are stored in the task flow definition itself, personalizations are saved in MDS. If you do not explicitly specify the MDS layer to be used to save personalizations, personalization metadata and application metadata may get stored in different layers.

  To ensure that personalizations are saved to the same layer as application metadata, you must clone the application's MDS session in the task flow's preference file and reference that while configuring personalization. For an example of how this is implemented, see "Example of a Framework Application Containing a Task Flow Created By Following the Guidelines"

D.2.4 Guidelines for Implementing Implicit Personalization

Users perform implicit personalizations while using the task flow itself, by way of resizing, sorting table columns, and so on. Personalization options in this case are not

invoked using dialogs and users may not even be aware that they are actually personalizing a page.

Adhere to the following guidelines while implementing implicit personalization:

- Do not overload the task flow with implicit personalization attributes. Select a few key, frequently used attributes to expose as implicit. Infrequently used or complex attributes can be handled with explicit personalization.
- Do not implement implicit personalization such that a dialog is invoked. Implicit personalization must be performed in the body of the task flow itself.

D.3 Example of a Framework Application Containing a Task Flow Created By Following the Guidelines

As a companion to this document, a sample application was developed using Oracle WebCenter Portal: Framework. This sample application, called `RecentPages`, is used as an example to illustrate how you can build a near-perfect task flow by adhering to the guidelines in this document. The `RecentPages` application contains a customizable page, `Welcome.jspx`, which contains a task flow named `recent-pages-taskflow-definition` created by following all the required guidelines.

This section only provides a high-level description of how the task flow was implemented. For detailed information about using task flows and Composer, see the following guides:

- *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*, for information about creating task flows.
- *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*, for information about editing, customizing, and personalizing task flows in Composer.

The `RecentPages` sample application is available for download from the Oracle WebCenter Portal 11g Demonstrations and Samples page on OTN at:

`http://webcenter.oracle.com`

The `RecentPages` application contains the following projects:

- A Model project
- A Portal project named `RecentPagesTaskFlow` that contains the `recent-pages-task-flow-definition` task flow
- Another Portal project named `WebPages` that contains the JSPX page in which the task flow is consumed

D.3.1 The RecentPagesTaskFlow Project

This project contains the task flow created by following the guidelines. The files of significance in this project are as follows:

- The `recent-pages-task-flow-definition.xml` file
- The `mainView.jsff` fragment
- The `simpleEditPopupView.jspx` page
- The `advancedEditPopupView.jspx` page
- The `recentPagesBean` Java bean

- The `Preference` Java bean

D.3.1.1 The recent-pages-task-flow-definition.xml File

This is an Oracle ADF task flow definition containing three views: `mainView`, `simpleEditPopupView`, and `advancedEditPopupView`. The task flow is designed such that `mainView` displays a list of recently created pages in the application, `simpleEditPopupView` displays a personalization option to all users, and `advancedEditPopupView` displays customization options to users with Edit or Customize permission.

The following table breaks down the code in `recent-pages-task-flow-definition.xml` and explains each snippet.

The outcomes from `mainView` to `simpleEditPopupView` and `advancedEditPopupView` are called `dialog:simpleEditPopupview` and `dialog:advancedEditPopupView` respectively. This is to ensure that the views open in separate dialogs, not inline. In addition, this task flow provides Refresh, Previous, and Next outcomes on `mainView`. The `simpleEditPopupView` and `advancedEditPopupView` views provide Save and Cancel outcomes.

The following table describes significant sections of code in the `recent-pages-task-flow-definition.xml` file.

Table D–2 Sections Of Code in the Task Flow Definition File

Code	Description
`<default-activity>mainView</default-activity>`	Defining the default view for the task flow: `mainView`

`<managed-bean>`
`  <managed-bean-name>recentPagesBean</managed-bean-name>`
`  <managed-bean-class>view.RecentPangesBean</managed-bean-class>`
`  <managed-bean-scope>pageFlow</managed-bean-scope>`
`</managed-bean>`
	Referencing the `recentPagesBean` Java bean that is associated with the task flow definition. This bean was created as part of the `view` package.

`<view id="mainView">`
`  <page>/mainView.jsff</page>`
`</view>`
`  <view id="simpleEditPopupView">`
`  <page>/simpleEditPopupView.jspx</page>`
`</view>`
`  <view id="advancedEditPopupView">`
`  <page>/advancedEditPopupView.jspx</page>`
`</view>`
	Mapping task flow views to pages or page fragments. For example, `mainView` is associated with `mainView.jsff`.

Table D–2 (Cont.) Sections Of Code in the Task Flow Definition File

Code	Description

`<control-flow-rule>`
`  <from-activity-id>mainView</from-activity-id>`
`  <control-flow-case>`
`  <from-outcome>refresh</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>dialog:simpleEditPoup</from-outcome>`
`  <to-activity-id>simpleEditPopupView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>next</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>previous</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>dialog:advancedEditPopup</from-outcome>`
`  <to-activity-id>advancedEditPopupView</to-activity-id>`
`  </control-flow-case>`
`</control-flow-rule>`
	Control flow rules from `mainView`. Each control flow case defines an outcome and a corresponding target view.

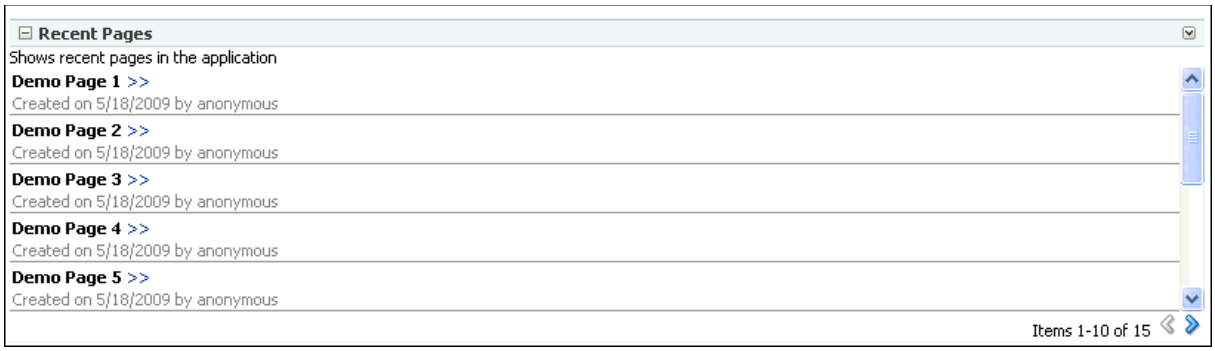
`<control-flow-rule>`
`  <from-activity-id>simpleEditPopupView</from-activity-id>`
`  <control-flow-case>`
`  <from-outcome>save</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>cancel</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`</control-flow-rule>`
	Control flow rules from `simpleEditPopupView`. Each control flow case defines an outcome and a corresponding target view.

`<control-flow-rule>`
`  <from-activity-id>advancedEditPopupView</from-activity-id>`
`  <control-flow-case>`
`  <from-outcome>save</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`  <control-flow-case>`
`  <from-outcome>cancel</from-outcome>`
`  <to-activity-id>mainView</to-activity-id>`
`  </control-flow-case>`
`</control-flow-rule>`
	Control flow rules from `advancedEditPopupView`. Each control flow case defines an outcome and a corresponding target view.

D.3.1.2 The mainView.jsff Fragment

Figure D–5 displays a list of recently created pages in the application. At runtime, the list looks like this:

Figure D–5 The mainView Fragment in the Task Flow



The design features of this view fragment are as follows:

- The `autoHeightRows` attribute is defined on the table to control the number of items displayed in the table. The task flow then wraps to the height of the items displayed.
- The `fetchSize` attribute is defined to control the number of items to be fetched from the data source and displayed in the table.
- Pagination controls are included to display a message in the format "`Items 1-4 of 25`" along with previous and next icons.

Table D–3 describes significant sections of code in the `mainView.jsff` fragment.

Table D–3 The mainView Fragment Code

Code	Description
`<af:panelGroupLayout id="pgl1" layout="vertical">` `  <af:outputText value="Shows recent pages in the application" id="ot1"/>`	A single top-level component-`Panel Group Layout`.This component is designed to ensure that the task flow content flows vertically and always stretches horizontally. This allows geometry management.

Table D–3 (Cont.) The mainView Fragment Code

Code	Description
<af:table value="#{pageFlowScope.recentPagesBean.pages}" var="row" rowBandingInterval="0" id="t1" rowSelection="none" columnBandingInterval="0" width="100%" inlineStyle="border:none;width:100%;" columnStretching="last" horizontalGridVisible="true" verticalGridVisible="false" **autoHeightRows="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}"** contentDelivery="immediate" **fetchSize="#{pageFlowScope.recentPagesBean.number_of_items_to_query}">**	A table component that is a child of the `Panel Group Layout`. Table attributes: ▪ `autoHeightRows` (`N`): The number of items to display in the table at one time. An EL value is specified here such that a Java bean, `recentPagesBean`, populates this attribute with a value specified by the user at runtime. If there are more than `N` items to display, a scrollbar appears on the task flow. ▪ `fetchSize` (`M`): The maximum number of items to be fetched from the data source. An EL value is specified here such that a Java bean, `recentPagesBean`, populates this attribute with a value specified by the user at runtime. If `M` is greater than `N`, pagination controls appear on the task flow to enable users to view the remaining items.
<af:column sortable="false" headerText="#{null}" align="start" id="c1"> <af:panelGroupLayout id="pgl11" styleClass="nowrap" inlineStyle="padding-top:2px; font-weight:bold;"> <af:outputText value="#{row.title}" id="ot4"/> <af:goLink destination="/faces#{row.contentMRef}" text=">>" targetFrame="_blank" id="gl1"/> </af:panelGroupLayout> <af:panelGroupLayout id="pgl10" styleClass="AFFieldTextDisabled nowrap" inlineStyle="padding-top:2px"> <af:outputText value="Created on " id="ot7"/> <af:outputText value="#{row.createDate}" id="ot8"/> <af:outputText value=" by #{row.createdBy}" id="ot9"/> </af:panelGroupLayout> </af:column>	The table contains a single column that displays a list of pages in the application along with the creation date and author for each page. Implicit personalization capability to sort the table column is turned off.

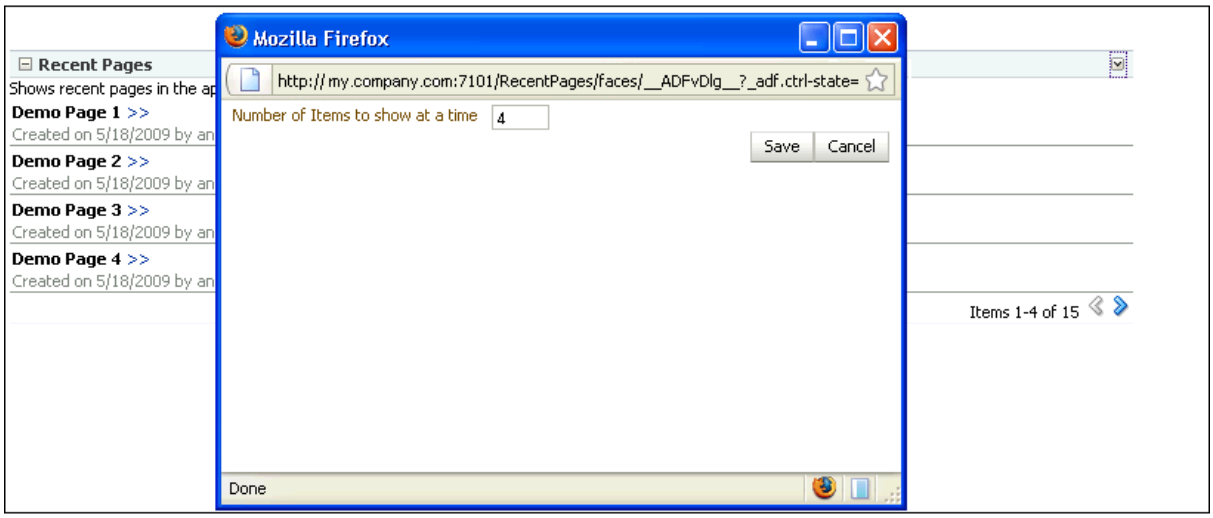
Table D–3 (Cont.) The mainView Fragment Code

Code	Description
<af:panelGroupLayout id="pgl2" halign="end" layout="vertical" inlineStyle="margin:1px" rendered="#{pageFlowScope.recentPagesBean.showPageControl}"> <af:panelGroupLayout id="pgl3"> <af:outputText value="#{pageFlowScope.recentPagesBean.footerText}" id="ot2"/> <af:commandLink id="cl1" actionListener="#{pageFlowScope.recentPagesBean.previous}" disabled="#{not pageFlowScope.recentPagesBean.showPrevious}" action="previous"> <af:image source="#{pageFlowScope.recentPagesBean.showPrevious ? '/adf/webcenter/shuttleleft_ena.png' : '/adf/webcenter/shuttleleft_dis.png'}" id="i1" shortDesc="previous"/> </af:commandLink> <af:commandLink id="cl2" actionListener="#{pageFlowScope.recentPagesBean.next}" disabled="#{not pageFlowScope.recentPagesBean.showNext}" action="next"> <af:image source="#{pageFlowScope.recentPagesBean.showNext ? '/adf/webcenter/shuttleright_ena.png' : '/adf/webcenter/shuttleright_dis.png'}" id="i2" shortDesc="next"/> </af:commandLink> </af:panelGroupLayout> </af:panelGroupLayout>	Pagination controls to display records beyond those that are displayed in the table. The `recentPagesBean` code defines what is displayed to users at runtime.

D.3.1.3 The simpleEditPopupView.jspx Page

Figure D–6 displays an option to select the number of items to be displayed in the task flow. All users viewing the page can personalize the task flow using this option. The value specified by a user is applicable only for that user's view of the page.

Figure D–6 The simpleEditPopupView Page



The following table describes significant sections of code in the `simpleEditPopupView.jspx` page.

Table D–4 Code in the Personalization Page

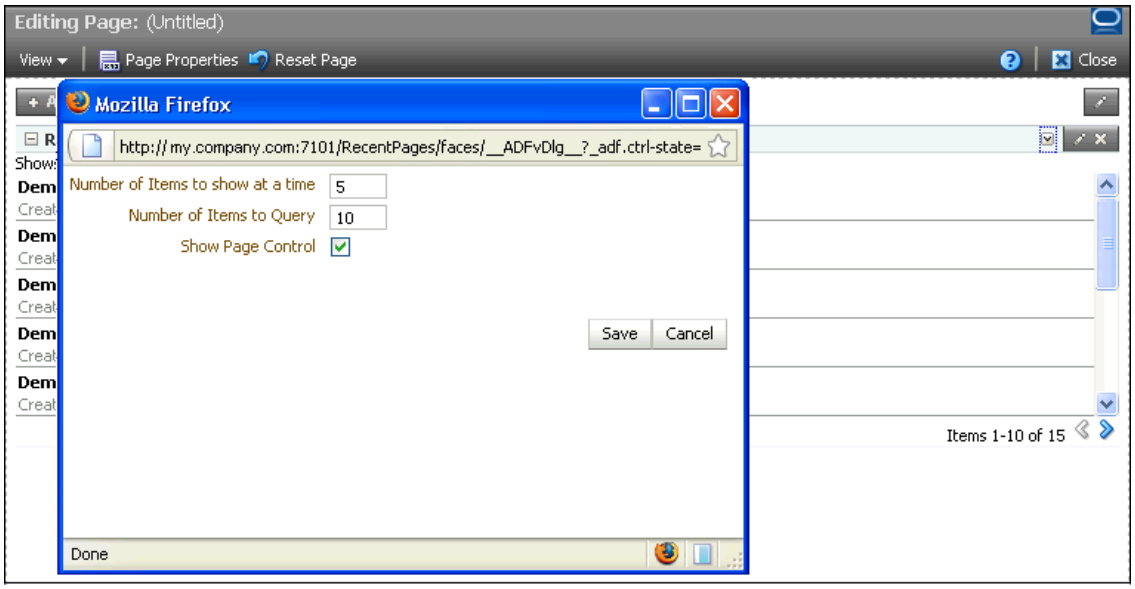
Code	Description

`<af:panelStretchLayout id="psl1">`
`  <f:facet name="center">`
`  <af:group id="group1">`
`  <af:panelFormLayout id="pfl1">`
`  <af:inputText label="Number of Items to show at a time"`
`  id="it1"`
`  value="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}"`
`  columns="3"/>`
`  </af:panelFormLayout>`
`  <af:panelGroupLayout id="pgl2"`
`  layout="horizontal"`
`  halign="right"`
`  inlineStyle="width:500px;">`
`  <af:commandButton text="Save" id="cb1"`
`  action="#{pageFlowScope.recentPagesBean.saveSettings}"/>`
`  <af:commandButton text="Cancel" id="cb2"`
`  action="cancel"`
`  actionListener="#{pageFlowScope.recentPagesBean.cancel}"/>`
`  </af:panelGroupLayout>`
`  </af:group>`
`  </f:facet>`
`</af:panelStretchLayout>`
	A single top-level `Panel Stretch Layout` component with a `Group` component in its center facet. Within this group are a `Panel Form Layout` and a `Panel Group Layout`.
	The `Panel Form Layout` contains an `Input Text` component, which is populated with a value returned by `recentPagesBean`.
	The `Panel Group Layout` contains two `Command Button` components called Save and Close, which are wired to events defined in `recentPagesBean`.

D.3.1.4 The advancedEditPopupView.jspx Page

This page displays customization options to users with Edit or Customize permission. At runtime, this page opens in a separate dialog and displays options to select the

number of items to be displayed and number of items to be queried, and a check box to show or hide pagination controls.



The following table describes significant sections of code in the `advancedPopupView.jspx` page.

Table D–5 Code in the Customization Page

Code	Description
<af:panelStretchLayout id="psl1"> <f:facet name="center"> <af:group id="group1"> <af:panelFormLayout id="pfl1"> <af:inputText label="Number of Items to show at a time" id="it1" value="#{pageFlowScope.recentPagesBean.number_of_items_to_show_at_a_time}" columns="3"/> <af:inputText label="Number of Items to Query" id="it2" value="#{pageFlowScope.recentPagesBean.number_of_items_to_query}" columns="3"/> <af:selectBooleanCheckbox label="Show Page Control" id="sbc1" value="#{pageFlowScope.recentPagesBean.showPageControl}"/> </af:panelFormLayout> <af:panelGroupLayout id="pgl2" layout="horizontal" halign="right" inlineStyle="width:500px;"> <af:commandButton text="Save" id="cb1" action="#{pageFlowScope.recentPagesBean.saveSettings}"/> <af:commandButton text="Cancel" id="cb2" action="cancel" actionListener="#{pageFlowScope.recentPagesBean.cancel}"/> </af:panelGroupLayout> </af:group> </f:facet> </af:panelStretchLayout>	A single top-level `Panel Stretch Layout` component with a group component in its center facet. Within this group are a `Panel Form Layout` and a `Panel Group Layout`. The `Panel Form Layout` contains two `Input Text` components and a `Select Boolean Checkbox`. These components are populated with values returned by `recentPagesBean`. The `Panel Group Layout` contains two `Command Button` components called Save and Close, which are wired to events defined in `recentPagesBean`.

D.3.1.5 The recentPagesBean Java Bean

This Java bean contains the logic for the following:

- Displaying a specific number of items in the table based on user input.
- Handling the Save and Cancel events triggered from the `simpleEditPopupView` and `advancedEditPopupView` pages.
- Rendering Previous and Next icons while implementing pagination.
- Displaying the pagination message after retrieving the total number of pages.
- Referencing the `Preference` bean that defines where customizations and personalizations must be saved.

You can find the code for this bean in the `RecentPages` application that is available for download from the Oracle WebCenter Portal 11g Demonstrations and Samples page on OTN at:

http://webcenter.oracle.com

The following table describes significant sections of code in the `recentPagesBean.java` file.

Table D–6 The recentPagesBean Java Bean Code

Code	Description

`private String _getPreferenceFileName()`
`  {`
`  String taskflowId =`
 	
`ControllerContextFwk.getInstanceFwk().getCurrentVi`
`ewPortFwk().getClientId();`
`  if (taskflowId == null)`
`  {`
`  taskflowId =`
 	
`ControllerContextFwk.getInstanceFwk().getCurrentVi`
`ewPortFwk().getParentViewPort().getClientId();`
`  }`
`  String preferenceFileName =`
`  PREFERENCE_BASE + taskflowId.replace(".", "_`
`") + ".xml";`
`  return preferenceFileName;`
`  }`
`  private static PageServiceBean`
`getPageServiceBean()`
`  {`
`  String beanName = "pageServiceBean";`
`  Object bean = null;`
`  FacesContext ftx =`
`FacesContext.getCurrentInstance();`
`  bean =`
`ftx.getExternalContext().getRequestMap().get(beanN`
`ame);`
`  if (bean == null)`
`  {`
`  ELContext elContext = ftx.getELContext();`
`  ExpressionFactory exFactory =`
 	
`ftx.getApplication().getExpressionFactory();`
`  ValueExpression expr =`
`  exFactory.createValueExpression(elContext,`
`"#{" + beanName + "}",PageServiceBean.class);`
`  bean = expr.getValue(elContext);`
`  }`
`  return (PageServiceBean) bean;`
` }
	Logic for identifying the task flow and retrieving the instance-level preference file for the currently running task flow.

Table D–6 (Cont.) The recentPagesBean Java Bean Code

Code	Description

`public String saveSettings()  throws Exception`
`  {`
`    if (number_of_items_to_query != null &&`
`        !number_of_items_to_query.isEmpty())`
`    {`
`      Preference.setNumberOfItemsToQuery(_`
`getPreferenceFileName(),`
 	
`Integer.parseInt(number_of_items_to_query));`
`    }`
`    if (this.number_of_items_to_show_at_a_time !=`
`null &&`
`        !this.number_of_items_to_show_at_a_`
`time.isEmpty())`
`    {`
`      Preference.setNumberOfItemsToShow(_`
`getPreferenceFileName(),`
 	
`Integer.parseInt(number_of_items_to_show_at_a_`
`time));`
`    }`
`    if (_showPageControl != null)`
`    {`
`      Preference.setShowPageControl(_`
`getPreferenceFileName(),_showPageControl);`
`    }`
 	
`AdfFacesContext.getCurrentInstance().returnFromDia`
`log(null, null);`
`    return "save";`
`  }`
	Logic to define behavior of Save button in the personalization or customization dialog. Settings are saved in the task flow preference file.

`public void setNumber_of_items_to_show_at_a_`
`time(String number_of_items_to_show_at_a_time)`
` {`
`    this.number_of_items_to_show_at_a_time =`
`       number_of_items_to_show_at_a_time;`
`  }`
`  public String getNumber_of_items_to_show_at_a_`
`time()`
`  {`
`    String query =`
 	
`Integer.toString(Preference.getNumberOfItemsToShow`
`(_getPreferenceFileName()));`
`    return query;`
`  }`
	Logic to populate the `Number of Items to show at a time` field in the personalization or customization dialog.

Table D–6 (Cont.) The recentPagesBean Java Bean Code

Code	Description

`public void setNumber_of_items_to_query(String number_of_items_to_query)`
`  {`
`  this.number_of_items_to_query = number_of_items_to_query;`
`  }`
`  public String getNumber_of_items_to_query()`
`  {`
`  String query =`
 	
`Integer.toString(Preference.getNumberOfItemsToQuery(_getPreferenceFileName()));`
`  return query;`
`  }`
	Logic to populate the `Number of Items to Query` field in the customization dialog.

`public boolean getShowPageControl()`
`  {`
`  boolean showPageControl = Preference.getShowPageControl(_getPreferenceFileName());`
`  return showPageControl;`
`  }`
`  public void setShowPageControl(boolean showPageControl)`
`  {`
`  _showPageControl = showPageControl;`
`  }`
	Logic to set the `Show Page Control` check box in the customization dialog.

`public List<PageDef> getPages()`
`{`
`  PageServiceBean psb = getPageServiceBean();`
`  List<PageDef> pages = psb.getPages();`
`  int start = getStart();`
`  int end = getEnd();`
`  if (start < 0 || start > pages.size())`
`  {`
`  start = 0;`
`  }`
`  if (end < 0 || end > pages.size())`
`  {`
`  end = pages.size();`
`  }`
`  List<PageDef> tempPages = pages.subList(start, end);`
`  return tempPages;`
`}`
`public int getEnd()`
`{`
`  String items_to_query = this.getNumber_of_items_to_query();`
`  int offset = Integer.parseInt(items_to_query);`
`  int end = getStart() + offset;`
`  if (end > this.getTableSize())`
`  {`
`  end = getTableSize();`
`  }`
`  return end;`
`}`
	Logic to retrieve the list of pages to display in the current view of the table.

Table D–6 (Cont.) The recentPagesBean Java Bean Code

Code	Description

`public int getTableSize()`
`{`
`  int tableSize = -1;`
`  try`
`  {`
`  PageServiceBean psb = getPageServiceBean();`
`  List<PageDef> pages = psb.getPages();`
`  tableSize = pages.size();`
`  }`
`  catch (Exception e)`
`  {`
`  System.out.println(e.toString());`
`  }`
`  return tableSize;`
`}`
	Logic to set the table size to the number of records retrieved.

`public int getStart()`
`{`
`  String items_to_query =`
`this.getNumber_of_items_to_query();`
`  int offset = Integer.parseInt(items_to_query);`
`  return (getPageNumber() - 1) * offset;`
`}`
`public int getPageNumber()`
`{`
`  Object page_num =`
 	
`RequestContext.getCurrentInstance().getPageFlowSco`
`pe().get("page_num");`
`  int local_page_num = 1;`
`  if (page_num == null || (Integer) page_num <= 0)`
`  {`
`  local_page_num = 1;`
`  }`
`  else`
`  {`
`  local_page_num = (Integer) page_num;`
`  }`
`  return local_page_num;`
`  }`	

`public void cancel(ActionEvent actionEvent)`
`  {`
 	
`AdfFacesContext.getCurrentInstance().returnFromDia`
`log(null, null);`
`  }`
	Logic to define behavior of Cancel button in the personalization or customization dialog.

Table D–6 (Cont.) The recentPagesBean Java Bean Code

Code	Description

`public void previous(ActionEvent actionEvent)`
`  {`
`  int page_num = getPageNumber();`
`  page_num--;`
`  if (page_num <= 0)`
`  {`
`  page_num = 1;`
`  }`
 	
`RequestContext.getCurrentInstance().getPageFlowSco`
`pe().put("page_num", page_num);`
`  }`
	Logic to define behavior of Previous action in the task flow's pagination control.

`public void next(ActionEvent actionEvent)`
`{`
`  int page_num = getPageNumber();`
`  page_num++;`
`  String items_to_query =`
`this.getNumber_of_items_to_query();`
`  int offset = Integer.parseInt(items_to_query);`
`  int total_page_num = getTotalPageNumber();`
`  if (page_num > total_page_num)`
`  {`
`  page_num = this.getTableSize()`
`/ offset;`
`  }`
 	
`RequestContext.getCurrentInstance().getPageFlowSco`
`pe().put("page_num",  page_num);`
`  }`
	Logic to define behavior of Next action in the task flow's pagination control.

`public boolean isShowPrevious()`
`  {`
`  int page_num = getPageNumber();`
`  if (page_num <= 1)`
`  {`
`  return false;`
`  }`
`  else`
`  {`
`  return true;`
`  }`
`  }`
	Logic to render Previous icon

`public boolean isShowNext()`
`  {`
`  int page_num = getPageNumber();`
`  if (page_num >= getTotalPageNumber())`
`  {`
`  return false;`
`  }`
`  else`
`  {`
`  return true;`
`  }`
`  }`
	Logic to render Next icon.

Table D–6 (Cont.) The recentPagesBean Java Bean Code

Code	Description

`public int getTotalPageNumber()`
`  {`
`  String items_to_query =`
`  this.getNumber_of_items_to_query();`
`  int offset = Integer.parseInt(items_to_query);`
`  int total_page_num = this.getTableSize()`
`  / offset;`
`  if (getTableSize() % offset > 0)`
`  {`
`  total_page_num++;`
`  }`
`  return total_page_num;`
`  }`
	Logic to return the total number of pages.

`public String getFooterText()`
`  {`
`  return "Items " + (this.getStart() + 1) + "-"`
`+ this.getEnd() +`
`" of " + " " + this.getTableSize() + " ";`
`  }`
	Logic for returning pagination text in the format `Items 1-4 of 25`.

D.3.1.6 The Preference Bean

This Java bean is a wrapper that provides APIs to get and set attributes in the task flow preference file. It contains the logic for saving personalizations and customizations on this task flow instance in the same MDS layer as that used by the application. In this sample application, site-level customizations are stored in the `mds/mdssys/cust/site/site` directory. Therefore, the code in the `Preference` bean ensures that the application's MDS session is cloned and the task flow personalizations and customizations are stored in the same directory.

You can find the code for this bean in the `RecentPages` application that is available for download from the Oracle WebCenter Portal 11g Demonstrations and Samples page on OTN at:

`http://webcenter.oracle.com`

The following table describes significant sections of code in the `Preference.java` file.

Table D–7 The Preference Java Bean Code

Code	Description

`private static MDSSession _getClonedMDSSession()`
`  {`
`  MDSInstance instance = (MDSInstance)`
`ADFContext.getCurrent().getMDSInstanceAsObject();`
`  MDSSession mdsSession = (MDSSession)`
`ADFContext.getCurrent().getMDSSessionAsObject();`
`  mdsSession =`
` instance.createSession(mdsSession.getSessionOptions(),`
`mdsSession.getUserStateHandler());`
`  return mdsSession;`
`  }`
	Logic to clone the application's MDS session.

Table D–7 (Cont.) The Preference Java Bean Code

Code	Description
`private static MetadataObject _ readPreference(MDSSession mdsSession, String preferenceFileName) { MetadataObject mo = null; try { mo = mdsSession.getMutableMO(MOReference.create(preferenceFileName)); } catch (MetadataNotFoundException mnfe) { _LOG.warning("Creating a new preference file " + preferenceFileName); } catch (ReferenceException re) { _LOG.warning("Creating a new preference file " + preferenceFileName); } catch (Exception e) { throw new RuntimeException(e); } // preference file is not available create a new one if (mo == null) { try { DocumentBuilderFactory builderFactory = DocumentBuilderFactory.newInstance(); builderFactory.setNamespaceAware(true); builderFactory.setValidating(false); DocumentBuilder builder = builderFactory.newDocumentBuilder(); Document document = builder.parse(new ByteArrayInputStream(CONTENTS.getBytes())); mo = mdsSession.createMetadataObject(preferenceFileName, document); mdsSession.flushChanges(); mo = mdsSession.getMutableMO(MOReference.create(preferenceFileName)); } catch (Exception e) { throw new RuntimeException(e); } } return mo; }`	Logic to read the preference file in the MDS session or create one, if required.

Table D–7 (Cont.) The Preference Java Bean Code

Code	Description
`public static void setNumberOfItemsToShow(String preferenceFileName, int numberOfItemsToShow) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); mo.getDocument().getDocumentElement().setAttribute("numberOfItemsToShow", Integer.toString(numberOfItemsToShow)); _flushChanges(session); } public static int getNumberOfItemsToShow(String preferenceFileName) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); String show = mo.getDocument().getDocumentElement().getAttribute("numberOfItemsToShow"); return Integer.parseInt(show); }`	Logic to get and set the number of items to show in the table at a time.
`public static void setNumberOfItemsToQuery(String preferenceFileName, int _numberOfItemsToQuery) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); mo.getDocument().getDocumentElement().setAttribute("numberOfItemsToQuery", Integer.toString(_numberOfItemsToQuery)); _flushChanges(session); } public static int getNumberOfItemsToQuery(String preferenceFileName) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); String query = mo.getDocument().getDocumentElement().getAttribute("numberOfItemsToQuery"); return Integer.parseInt(query); }`	Logic to get and set the number of items to query from the data source.

Table D–7 (Cont.) The Preference Java Bean Code

Code	Description
`public static void setShowPageControl(String preferenceFileName, boolean showPageControl) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); mo.getDocument().getDocumentElement().setAttribute("showPageControl", Boolean.toString(showPageControl)); _flushChanges(session); } public static boolean getShowPageControl(String preferenceFileName) { MDSSession session = _getClonedMDSSession(); MetadataObject mo = _readPreference(session, preferenceFileName); String pageControl = mo.getDocument().getDocumentElement().getAttribute("showPageControl"); return Boolean.parseBoolean(pageControl); }`	Logic to get and set whether to show page control on the task flow.
`private static void _flushChanges(MDSSession session) { try { session.flushChanges(); } catch (Exception e) { throw new RuntimeException(e); } }`	Logic to save changes made to the task flow into the MDS session.If a sand box is set up in the MDS session, the changes are saved to the sand box. Else, they are permanently written to MDS.
`private static final String CONTENTS = "<preference numberOfItemsToShow='5' numberOfItemsToQuery='10' showPageControl='true' xmlns='http://xmlns.oracle.com/apps/preference'/>"; }`	Logic to display defaults in the Number of Items to show at a time and `Number of Items to query fields` and the `Show Page Control` check box. When a user first accesses an instance of the `RecentPages` task flow, a new preference file is created with this content.

The `RecentPages` project containing all these files is deployed to Oracle ADF and the task flow is then consumed in the customizable `Welcome.jspx` page in the `WebPages` project.

D.3.2 The WebPages Project

This project contains the `Welcome.jspx` page in which the `recent-pages-task-flow-definition` task flow is consumed.

D.3.2.1 The Welcome.jspx File

`Welcome.jspx` is a customizable page containing a `Panel Stretch Layout` component. The top facet of the `Panel Stretch Layout` contains a `Change Mode Link` and a Page Create task flow. The center facet contains a `Page Customizable` with a child `Panel Customizable`, which in turn contains a `Show Detail Frame` component with the `recent-pages-task-flow-definition` task flow.

The `stretchContent` and `showResizer` attributes on the `Show Detail Frame` are set to `false` and `never` respectively to ensure that this component does not stretch its child task flow.

Also, the `refresh` attribute is set to `ifNeeded` in the task flow binding. This is to ensure that customizations made to the page at runtime are reflected when users perform a refresh.

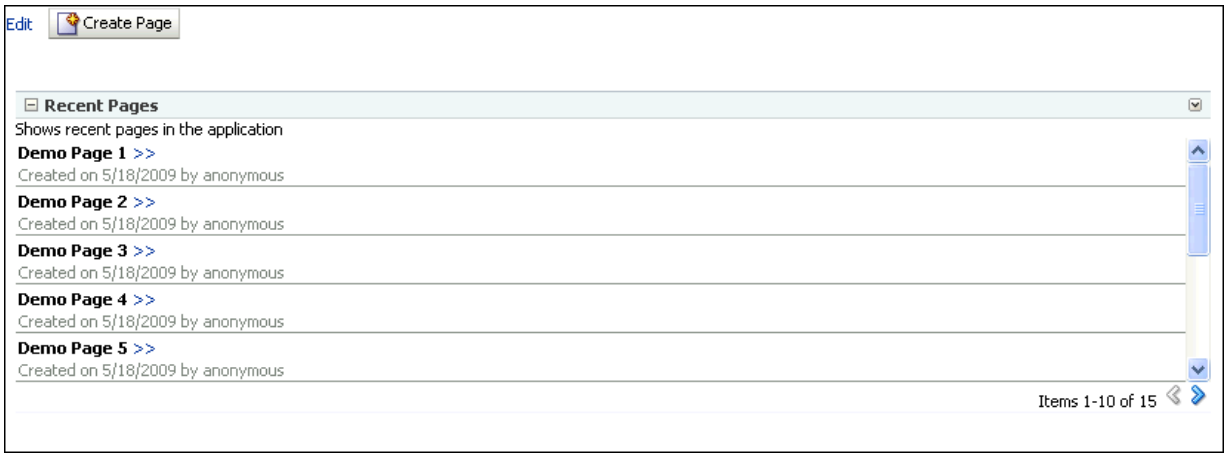
The `Welcome.jspx` page contains the following code:

```
<?xml version='1.0' encoding='US-ASCII'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:f="http://java.sun.com/jsf/core"
          xmlns:h="http://java.sun.com/jsf/html"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
          xmlns:pe="http://xmlns.oracle.com/adf/pageeditor"
          xmlns:cust="http://xmlns.oracle.com/adf/faces/customizable">
  <jsp:directive.page contentType="text/html;charset=US-ASCII"/>
  <f:view>
    <af:document id="d1">
      <af:form id="f1">
        <af:panelStretchLayout id="psl1">
          <f:facet name="top">
            <af:panelGroupLayout layout="horizontal"
      xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
                                 id="pgl2">
              <pe:changeModeLink id="cml1"/>
              <af:spacer width="10" height="10" id="s2"/>
              <af:region value="#{bindings.pagecreatepage1.regionModel}"
                         id="r1"/>
              <af:panelGroupLayout id="pgl1"/>
            </af:panelGroupLayout>
          </f:facet>
          <f:facet name="center">
            <pe:pageCustomizable id="pageCustomizable1">
              <cust:panelCustomizable id="panelCustomizable1" layout="scroll">
                <cust:showDetailFrame text="Recent Pages" id="sdf1"
                 stretchContent= "never" showResizer="never">
                  <af:region
value="#{bindings.recentpagestaskflowdefinition1.regionModel}"
                            id="r2"/>
                </cust:showDetailFrame>
              </cust:panelCustomizable>
              <f:facet name="editor">
                <pe:pageEditorPanel id="pep1"/>
              </f:facet>
            </pe:pageCustomizable>
          </f:facet>
          <f:facet name="start"/>
          <f:facet name="end"/>
        </af:panelStretchLayout>
      </af:form>
    </af:document>
  </f:view>
```

```
</jsp:root>
```

At runtime, the page appears as shown in Figure D–7.

Figure D–7 The Welcome.jspx Page



D.3.2.2 The adf-config.xml File

You can add a task flow inside a `Show Detail Frame` component and display the task flow's outcomes as actions on the chrome of the parent `Show Detail Frame` component. This is possible by defining custom actions on the parent `Show Detail Frame` component. You can define custom actions at the instance level where `Custom Action` components are added as child components of the `Show Detail Frame`, or at the global level where `Custom Action` elements are defined in the application's `adf-config.xml` file. Global custom actions are enabled on all `Show Detail Frame` instances in the application, but are displayed only when a `Show Detail Frame` instance contains the task flow as its child.

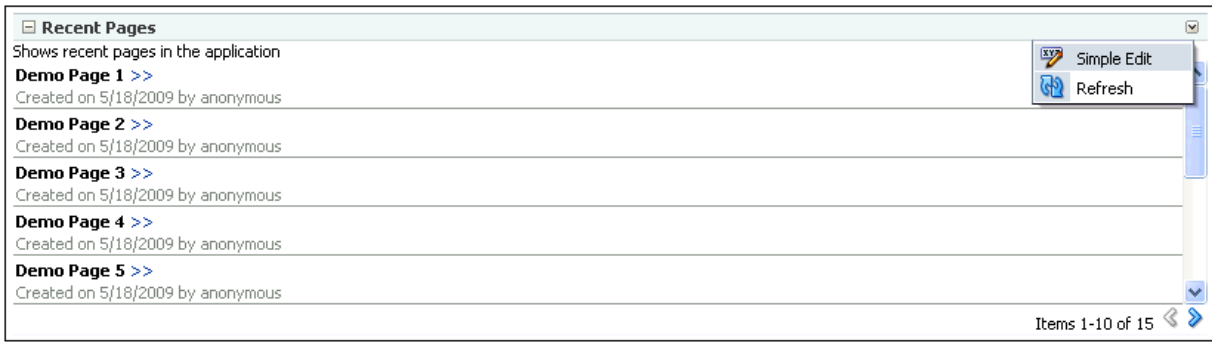
In this application, custom actions corresponding to the task flow's outcomes, `dialog:simpleEditPopupView` and `dialog:advancedEditPopupView`, are defined globally in the `adf-config.xml` file as follows:

```
<customizableComponentsSecurity
xmlns="http://xmlns.oracle.com/adf/faces/customizable/config">
    <enableSecurity value="true"/>
    <customActions>
      <customAction action="refresh" text="Refresh"
                    shortDesc="Refresh"
                    location="menu" rendered="true"
                    icon="/adf/pe/images/refresh_ena.png"/>
      <customAction action="dialog:simpleEditPoup"
                    text="Simple Edit"
                    shortDesc="Simple Edit"
                    location="menu"
                    rendered="#{!changeModeBean.inEditMode}"
                icon="/adf/pe/images/editproperties_ena.png"/>
      <customAction action="dialog:advancedEditPopup"
                    text="Advanced Edit"
                    shortDesc="Advanced Edit" location="menu"
                    rendered="#{changeModeBean.inEditMode}"
                icon="/adf/pe/images/editproperties_ena.png"/>
    </customActions>
  </customizableComponentsSecurity>
```

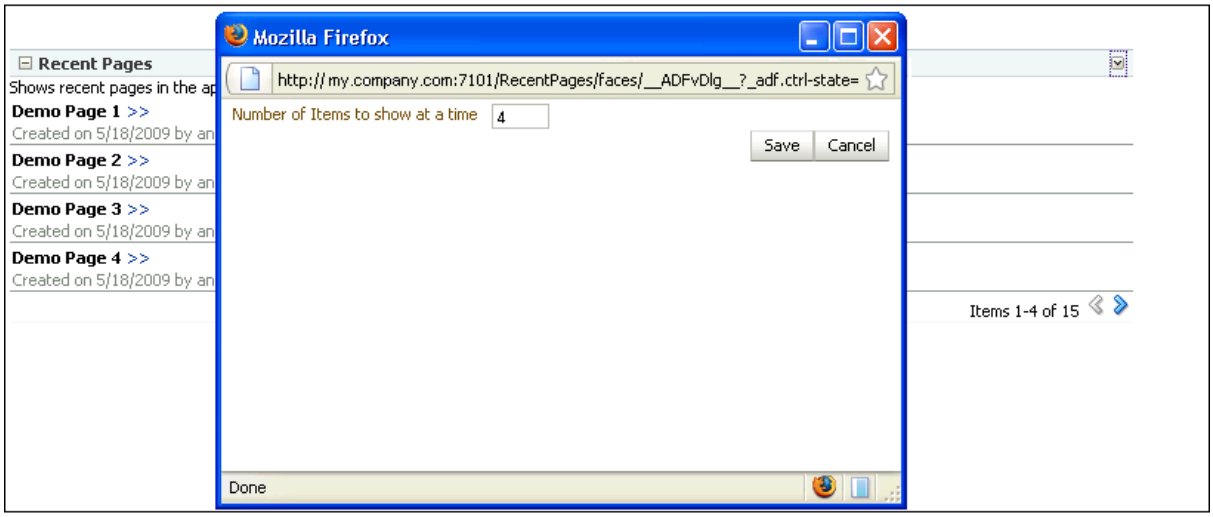
D.3.3 Runtime Behavior

This configuration results in the following behavior at runtime:

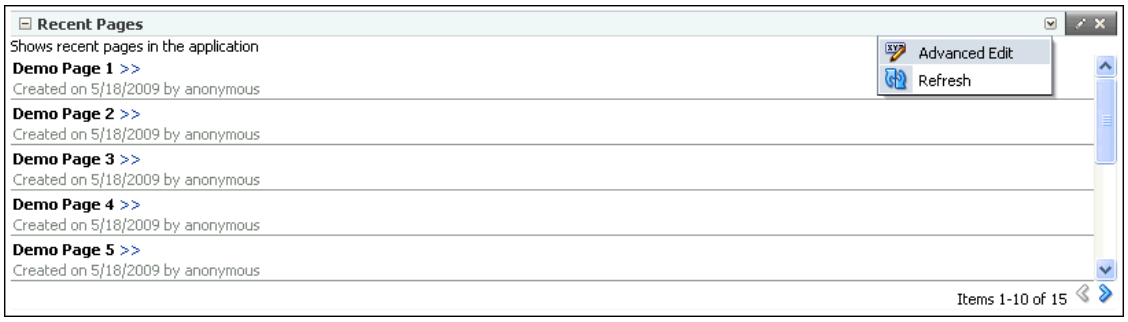
- All users see a Simple Edit action on the chrome surrounding the task flow while viewing the page.



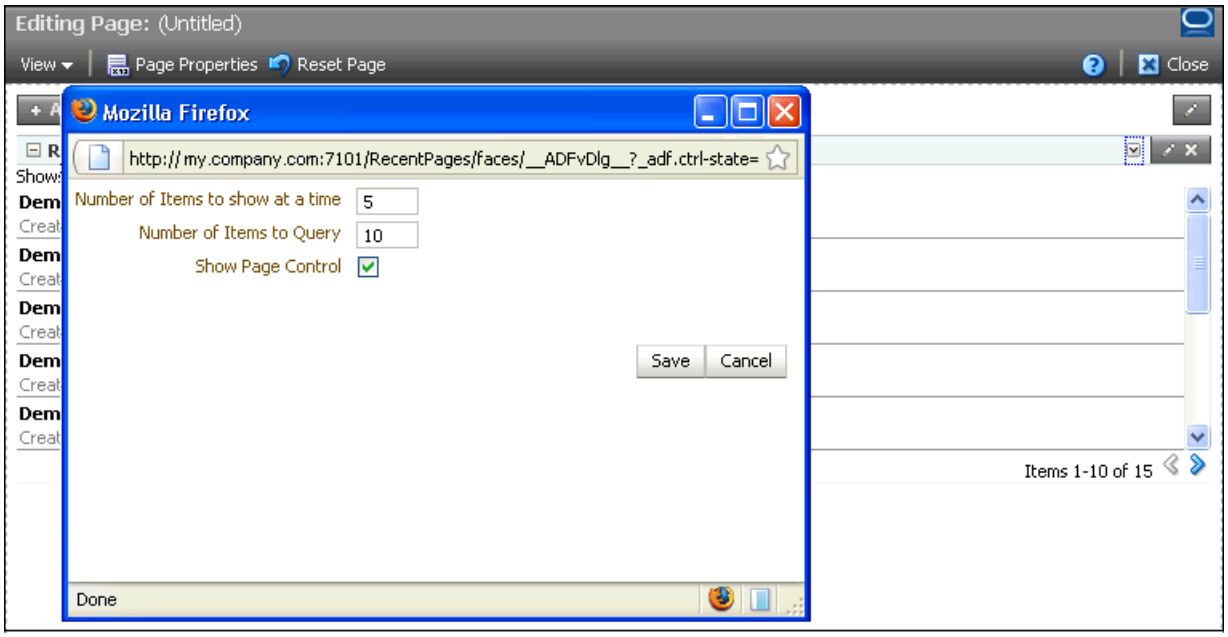
Clicking this action displays the personalization option in a separate dialog.



- Users with Edit or Customize permission see an Advanced Edit action on the chrome surrounding the task flow while editing the page.



Clicking this action displays customization options in a separate dialog.



D.4 Conclusion

In Framework application pages containing task flows, task flow design can have a significant impact on application performance. Also, slow response time and inconsistent UI can lead to an unpleasant experience for application users. To ensure better performance and a good user experience for Framework application users, follow the guidelines in this document while implementing task flows to be consumed in the application.

E

Additional Portlet Configuration

This appendix discusses configuration information for the portlet technologies available with Oracle WebCenter Portal.

This chapter includes the following sections:

- Section E.1, "OmniPortlet Configuration Tips"
- Section E.2, "Web Clipping Portlet Configuration Tips"
- Section E.3, "Hiding or Removing the WSRP Test Page"
- Section E.4, "Setting Up a Preference Store"
- Section E.5, "Portlet Preference Store Migration Utilities"
- Section E.6, "Exporting and Importing Portlet Producers at Design Time"
- Section E.7, "Migrating WebLogic Portal Portlets to WebCenter Portal"

For more information about using the portlet technologies at design time, see their respective chapters throughout this guide:

- Chapter 60, "Creating Portlets with the Portlet Wizard"
- Chapter 61, "Coding Portlets"
- Chapter 65, "Creating Portlets with OmniPortlet"
- Chapter 66, "Creating Content-Based Portlets with Web Clipping"

For more information about using these technologies at runtime, see their respective chapters in *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.

E.1 OmniPortlet Configuration Tips

This section contains configuration information for OmniPortlet. To learn more about the OmniPortlet wizard, see Chapter 65, "Creating Portlets with OmniPortlet." This section contains configuration information for the following areas:

- Section E.1.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall"
- Section E.1.2, "Configuring the OmniPortlet Producer to Access Other Relational Databases"
- Section E.1.3, "Configuring Portal Tools and Web Producers (Optional)"

Note: In this section, *OmniPortlet_WAR_DIR* indicates the directory where `omniPortlet.war` is deployed in the WebLogic Server. The exact path can vary depending on your installation. To determine this path, search for `omniPortlet/provider.xml` in the file system. For example, run the following command in UNIX:

```
> find DOMAIN_DIR -name "provider.xml" | grep -i omniportlet
```

In the Integrated WebLogic Server (Integrated WLS or Default Server), the path of OmniPortlet's `provider.xml` is located here:

JDEV_SYSTEM_DIRECTORY/DefaultDomain/servers/DefaultServer/tmp/_WL_user/portalTools_11.1.1.2.0/*RANDOMLY_GENERATED_DIRECTORY*/war/WEB-INF/providers/omniPortlet/provider.xml

Note that, on a Windows platform, pages in Framework applications are not rendered if there is a space in the path to the system directory in JDeveloper. Therefore, ensure that the *JDEV_SYSTEM_DIRECTORY* path does not contain spaces.

In the Fusion Middleware 11*g* installation, the path of OmniPortlet's `provider.xml` is located here:

FMW_HOME/user_projects/domains/wc_domain/servers/WLS_Portlet/tmp/_WL_user/portalTools_*version_number*/*RANDOMLY_GENERATED_DIRECTORY*/war/WEB-INF/providers/omniPortlet/provider.xml

E.1.1 Configuring the OmniPortlet Producer to Access Data Outside a Firewall

If the OmniPortlet producer is inside your firewall, then you must configure the proxy information so that OmniPortlet can access URLs of data (such as CSV, XML, or web services) located outside the firewall. To do so, you can either set the proxy information in the command line when you start your WebLogic server. Or, you can set up the proxy information in OmniPortlet's `provider.xml` file, located here: *OmniPortlet_WAR_DIR*`/WEB-INF/providers/omniPortlet/provider.xml`.

Note: For the Web Service data source, you must set the proxy information in both the `provider.xml` file and using the command line parameters.

- To set the proxy information in the command line when starting the WebLogic server, set the parameters as described in Table E–1, if you are using an HTTP Proxy Host, or Table E–2, if you are using an HTTPS Proxy Host.

Table E–1 HTTP Proxy Information Command Line Parameters

Parameter	Description
`http.proxyHost`	The host name of a proxy server if one is required to make a URL connection from the OmniPortlet producer to its data sources.
`http.proxyPort`	The port number for the HTTP Proxy Host.
`http.nonProxyHosts`	The name of any domain or hosts to which you can directly connect, bypassing a proxy server, such as your local machine: `localhost|localhost.localdomain` Hosts can be fully qualified host names or can be IP addresses.

Table E–1 (Cont.) HTTP Proxy Information Command Line Parameters

Parameter	Description
`http.proxyUser`	The user to log in to the proxy server if the proxy server requires authentication.
`http.proxyPassword`	The password to log in to the proxy server if the proxy server requires authentication.
`http.proxyAuthType`	The authentication type of the proxy server. Acceptable values: `Basic | Digest`.
`http.proxyAuthRealm`	The name of the realm of the proxy server. If you do not know the name of the realm, contact the proxy server administrator.

Table E–2 HTTPS Proxy Information Command Line Parameters

Parameter	Description
`https.proxyHost`	The host name of a proxy server if one is required to make a URL connection from the OmniPortlet producer to its data sources.
`https.proxyPort`	The port number for the HTTPS Proxy Host.
`https.nonProxyHosts`	The name of any domain or hosts to which you can directly connect, bypassing a proxy server, such as your local machine: `localhost|localhost.localdomain` Hosts can be fully qualified host names or can be IP addresses.
`https.proxyUser`	The user to log in to the proxy server if the proxy server requires authentication.
`https.proxyPassword`	The password to log in to the proxy server if the proxy server requires authentication.
`https.proxyAuthType`	The authentication type of the proxy server. Acceptable values: `Basic | Digest`.
`https.proxyAuthRealm`	The name of the realm of the proxy server. If you do not know the name of the realm, contact the proxy server administrator.

The following are examples of three parameters and their values:

```
-Dhttps.proxyHost=myProxyServer.mycompany.com
-Dhttps.proxyPort=80
-Dhttps.nonProxyHosts=localhost|localhost.localdomain|127.0.0.1|
```

- To configure the proxy information in the `provider.xml` file, see Table E–3 for a list of parameters and their descriptions.

Table E–3 Provider.xml Tags

Parameter	Description
`httpProxyHost`	Enter the host name of a proxy server if one is required to make a URL connection from the OmniPortlet producer to its data sources.
`httpProxyPort`	Enter the port number for the HTTP Proxy Host.

Table E–3 (Cont.) Provider.xml Tags

Parameter	Description
dontProxyFor	Enter the name of any domain or hosts to which you can directly connect, bypassing a proxy server. Domain names are the part of a URL that contain the names of a business, or organization, or government agency, for example: *.company.com, *.us.company.com Hosts can be fully qualified host names or can be IP addresses.
proxyUseAuth	Acceptable values: true | false Enter true if your proxy server requires authentication. The authentication parameters are specified by the following tags: proxyType, proxyRealm, proxyUseGlobal, proxyUserName, and proxyPassword.
proxyType	Acceptable values: Basic | Digest Choose the type of proxy server this provider. For more information about basic or digest authentication, see http://www.faqs.org/rfcs/rfc2617.html.
proxyRealm	Enter the name of the realm of the proxy server that is accessed by the user according to the login information described later in the table. If you do not know the name of the realm, then contact the administrator of the proxy server.
proxyUseGlobal	Acceptable values: true | false If true, then the <proxyUser> and <proxyPassword> values are used for all users. Users do not see the Proxy Authentication section on the Source tab and Personalize page. If false, then page designer must log in using the Proxy Authentication section on the Source tab when they define the portlet. The end user must log in using the Proxy Authentication section on the Personalize screen. If <proxyUsername> and <proxyPassword> are given, then they are only used for public users.
proxyUserName	Enter the user name to log in to the proxy server.
ProxyPassword	Enter the password for the specified user name. You must prefix ! before your plain password text. It is then encrypted in the provider.xml file for protection when the producer starts.

The following is a basic example of using a proxy to access data outside a firewall:

```
<proxyInfo class="oracle.portal.provider.v2.ProxyInformation">
<httpProxyHost>www-proxy.example.com</httpProxyHost>
<httpProxyPort>80</httpProxyPort>
<proxyUseAuth>false</proxyUseAuth>
</proxyInfo>
```

The following example requires a login and basic authentication for all users for the proxy server:

```
<proxyInfo class="mycompany.portal.provider.v2.ProxyInformation">
<httpProxyHost>myport.example.com</httpProxyHost>
<httpProxyPort>8080</httpProxyPort>
<proxyUseAuth>true</proxyUseAuth>
<proxyType>Basic</proxyType>
<proxyRealm>myport</proxyRealm>
<proxyUseGlobal>false</proxyUseGlobal>
</proxyInfo>
```

E.1.2 Configuring the OmniPortlet Producer to Access Other Relational Databases

The OmniPortlet SQL data source is preconfigured to access Oracle databases using the Oracle JDBC drivers, and ODBC data sources using Sun Microsystem's JDBC-ODBC driver. Oracle allows developers to access other relational databases using DataDirect JDBC drivers.

> **See Also:** For a list of supported databases, Certification Matrix for Oracle Application Server and DataDirect JDBC available on the Oracle Technology Network (http://www.oracle.com/technetwork/index.html).

This section contains the following steps:

- Section E.1.2.1, "Installing DataDirect JDBC Drivers"
- Section E.1.2.2, "Registering DataDirect Drivers in OmniPortlet"

E.1.2.1 Installing DataDirect JDBC Drivers

The following DataDirect JDBC drivers are included with the WebLogic Server installation:

- `wlinformix.jar`
- `wlsqlserver.jar`
- `wlutil.jar`
- `wldb2.jar`
- `wlresource.jar`
- `wlspy.jar`
- `wlbase.jar`

If you do not plan to use these DataDirect drivers, you can instead download DataDirect JDBC drivers to access the desired database. These drivers are packaged in a single ZIP, which you can download from the following location:

http://www.oracle.com/technetwork/topics/datadirect-index-091847.html

To install DataDirect JDBC drivers:

1. Unzip the contents of the ZIP file into a temporary directory, for example `/temp/datadirect`.
2. Copy the DataDirect JDBC drivers from the temporary directory to your WebLogic Server directory: `WLS_DOMAIN_DIRECTORY/lib`.

E.1.2.2 Registering DataDirect Drivers in OmniPortlet

OmniPortlet is implemented as a Web producer and all the configuration properties are stored in the `provider.xml` file. To use DataDirect JDBC drivers with OmniPortlet, you must register these drivers in the `provider.xml` file.

To register the new DataDirect JDBC drivers:

1. Back up the file, *`OmniPortlet_WAR_DIRECTORY`*`/WEB-INF/providers/omniPortlet/provider.xml`, and then open the file.
2. Add the drivers to use for the SQL data source configuration entry:

a. Search for the XML tag, `driverInfo`.

b. Add a new entry after the last `driverInfo` tag.

The following are examples of the driverInfo for WebLogic DataDirect drivers:

- Microsoft SQL Server:

```
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>Microsoft SQL Server</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>weblogic:sqlserver</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>weblogic.jdbc.sqlserver.SQLServerDriver
  </driverClassName>
  <dataSourceClassName>weblogic.jdbcx.sqlserver.SQLServerDataSource
  </dataSourceClassName>
  <connHandlerClass>oracle.webdb.reformlet.data.jdbc.JDBCConnectionHandler
  </connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

- Sybase:

```
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>Sybase</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>weblogic:sybase</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>weblogic.jdbc.sybase.SybaseDriver
  </driverClassName>
  <connHandlerClass>
  oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler
  </connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

- DB2:

```
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>DB2</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>weblogic:db2</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>weblogic.jdbc.db2.DB2Driver
  </driverClassName>
  <connHandlerClass>
  oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler
  </connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

- Informix:

```
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>Informix</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>weblogic:informix</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>weblogic.jdbc.informix.InformixDriver
```

```
  </driverClassName>
  <connHandlerClass>
  oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler
  </connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

Table E–4 describes the parameters in the `driverInfo` property.

Table E–4 Parameters in the driverInfo Property

Parameter	Description
`name`	Name of the database you want to use. This name is used on the Source tab of the OmniPortlet wizard.
`sourceDataBase`	Internal value. Set the value to `other`.
`subProtocol`	JDBC subprotocol name used by OmniPortlet to create the connection string, for example `sqlserver`, `sybase`, or `db2`. To get the list of subprotocol names, see the DataDirect JDBC driver documentation using the links provided at the end of this section.
`connectString`	Description of the connect string format. For DataDirect drivers, the format is: `mainProtocol:subProtocol://databaseName`
`driverClassName`	Name of the driver class. To get the different values, see the DataDirect JDBC driver documentation using the links provided at the end of this section.
`dataSourceClassName`	Name of the data source class that implements connection pooling. This parameter is only available in OmniPortlet version 9.0.4.1 or later. See Table E–5 for the right data source class name for your driver.

`connHandlerClass`	Class used by OmniPortlet to manage the driver and connection pooling. The value is either of the following:
	▪ For OmniPortlet version 9.0.4.1 or later:
	`oracle.webdb.reformlet.data.jdbc.JDBCConnectionHandler`
	▪ For OmniPortlet versions before 9.0.4.1:
	`oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler`

`connPoolSize`	Minimum number of connections that are opened by the connection pool.
`loginTimeOut`	Maximum time, in seconds, that this data source waits while attempting to connect to a database.

Table E–5 lists the values for the `driverClassName` and `dataSourceClassName` properties for specific DataDirect JDBC drivers.

Table E–5 Parameters and Values for driverClassName and dataSourceClassName

DataDirect Drivers Supported	Properties
Microsoft SQL Server	Parameter: `driverClassName` Value: `weblogic.jdbc.sqlserver.SQLServerDriver`

Table E–5 (Cont.) Parameters and Values for driverClassName and

DataDirect Drivers Supported	Properties
Sybase	Parameter: `driverClassName` Value: `weblogic.jdbc.sybase.SybaseDriver`
DB2	Parameter: `driverClassName` Value: `weblogic.jdbc.db2.DB2Driver`
Informix	Parameter: `driverClassName` Value: `weblogic.jdbc.informix.InformixDriver`

3. Save the `provider.xml` file.
4. Stop and start the Oracle WebLogic Managed Server instance where your portlet producer was deployed. To do so, navigate to your `WL_HOME`, then to the subdirectory `opmn/bin`.

Note: If you are using OmniPortlet in a multiple nodes configuration, for example, in a clustering or load-balancing environment, then you must manually copy the `provider.xml` file on each node.

See Also: For more information about how to use DataDirect JDBC drivers, see Chapter 65, "Creating Portlets with OmniPortlet."

E.1.3 Configuring Portal Tools and Web Producers (Optional)

To ensure that the OmniPortlet and Web Clipping producers, locally built, and custom built Web producers work properly in the middle-tier environment, some additional configuration may be needed. If OmniPortlet or any other Web producers have customizations in the file system, then PDK-Java provides a Preference Store Migration/Upgrade Utility that you can use to migrate the existing customizations to the database and upgrade customizations from earlier releases. See Section E.5, "Portlet Preference Store Migration Utilities" for more information about the PDK Preference Store Migration Utility.

Configuring Portal Tools Producers in the Multiple Middle-Tier Environment

By default, the OmniPortlet producer uses the Database Preference Store. It can work in a multiple middle-tier environment without additional configuration.

You can find more information about configuring the database Preference Store in Section E.5, "Portlet Preference Store Migration Utilities."

1. If you have created an OmniPortlet instance with customizations in the file system, then you must migrate these customizations to the database using the Preference Store Migration Utility.

   To run the migration utility:

   1. Navigate to the WebCenter Portal Oracle home directory using the following command:

      ```
      cd WC_ORACLE_HOME
      ```

2. Run the following command to migrate OmniPortlet data from a file-based Preference Store (`FilePreferenceStore`) to the database Preference Store (`DBPreferenceStore`):

```
java -classpath
lib/dms.jar:jdbc/lib/ojdbc14dms.jar:portal/jlib/pdkjava.jar:portal/jlib/
ptlshare.jar oracle.portal.provider.v2.preference.MigrationTool -mode
filetodb -pref1UseHashing true -pref1RootDirectory
portal/portletdata/tools/omniPortlet
-pref2User User_Name -pref2Password User_Password -pref2URL
jdbc:oracle:thin:@infra.host.com:1521:orcl
```

See Section E.5, "Portlet Preference Store Migration Utilities" for more information about the PDK Preference Store Migration Utility.

2. Typically, you perform the HTTP Proxy configuration for OmniPortlet and Web Clipping before you configure the Load Balancing Router (LBR). To do it after the LBR is configured, perform the following steps:

   a. The Portal Tools configuration information is stored in the `provider.xml` file on the middle-tier server. You must update the configuration directly on one middle tier (for example, **M1**) and then propagate it across all middle tiers front-ended by the LBR. You must first shut down all middle tiers except **M1**.

   b. You can change the HTTP Proxy settings in the `provider.xml` file. For more information, see Section E.1.1, "Configuring the OmniPortlet Producer to Access Data Outside a Firewall."

   c. Propagate the changes made to the `provider.xml` file to middle tier **M2**:

      – Copy *OmniPortlet_WAR_DIR*`/WEB-INF/providers/omniPortlet/provider.xml` from **M1** to **M2**.

      – Copy *WebClipping_WAR_DIR*`/WEB-INF/providers/webClipping/provider.xml` from **M1** to **M2**.

3. Restart middle tier **M2**.

4. Update portlet producer registration in your Framework application. Change the first part of the producer registration URL from `http://m1.abc.com:7777/` to `http://lbr.abc.com/`.

5. Verify that OmniPortlet and the Web Clipping producers work properly through the LBR, by going to the test pages at the following URLs.

   - OmniPortlet Producer:
     `http://lbr.abc.com/portalTools/omniPortlet/producers/omniPortlet`

     If you see the "No Portlets Available" message under the Portlet Information section in the OmniPortlet Producer test page, then you may not have configured OmniPortlet correctly in Step 1. If OmniPortlet is configured correctly, then the OmniPortlet and Simple Parameter Form portlets are available on the test page.

   - Web Clipping Producer:
     `http://lbr.abc.com/portalTools/webClipping/producers/webClipping`

Note: To use the Web Clipping portlet, or the Web Page Data Source for OmniPortlet, then you must also enable session binding in Oracle Web Cache.

E.2 Web Clipping Portlet Configuration Tips

Before you use Web Clipping, you must perform some administrative tasks, including the following:

- Section E.2.1, "Web Clipping Repository Configuration"
- Section E.2.2, "HTTP or HTTPS Proxy Configuration"
- Section E.2.3, "Web Clipping Producer Security"

E.2.1 Web Clipping Repository Configuration

Web clippings have definitions that must be stored persistently in an Oracle Metadata Services (MDS) store or an Oracle database.

Note: You cannot use a Microsoft SQL Server or IBM DB2 database as the Web Clipping repository.

You can view the Web Clipping repository configuration by accessing the Web Clipping Producer Test Page at:

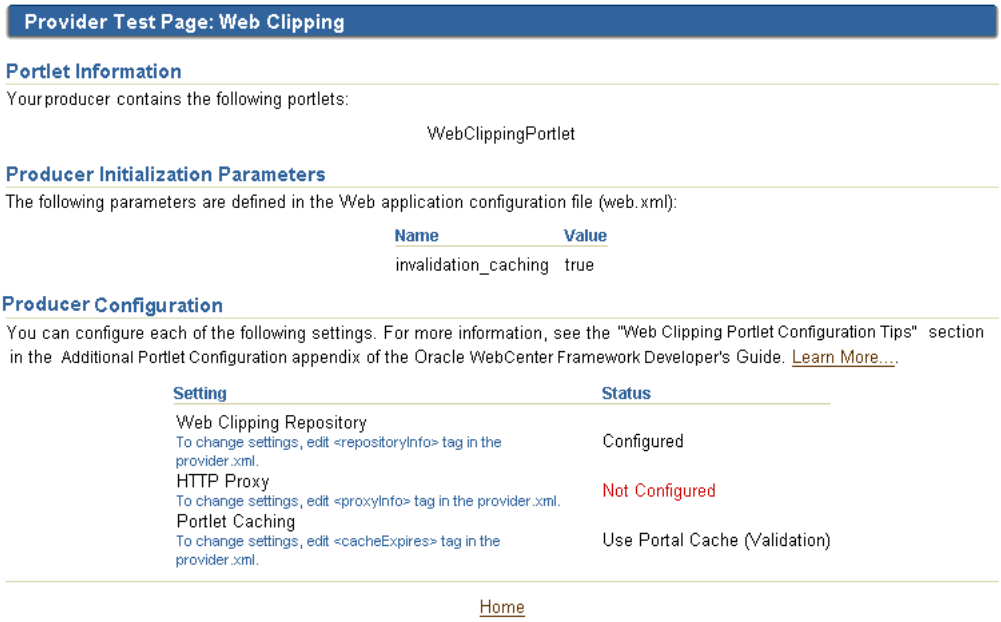
`http://host:port/portalTools/webClipping/providers/webClipping`

Where, *`host`* is the server to which your Web Clipping producer has been deployed and *`port`* is the port to which the server is listening for HTTP requests.

Note: Integrated WLS and the `WLS_Portlet` managed server are deployed to different ports even if they may be available on the same system. By default, Integrated WLS is deployed to port 7101 and `WLS_Portlet` is deployed to port 8889.

The Provider Test Page automatically detects whether the Web Clipping producer is configured with a valid repository. If it is not, then the **Status** column for the Web Clipping Repository displays **Not Configured**. Figure E–1 shows the Provider Test Page of Web Clipping.

Figure E–1 Web Clipping - Provider Test Page



You cannot change the Web Clipping configuration information by using the Provider Test Page. You can configure the Web Clipping repository by setting appropriate values in the `provider.xml` file. In this file, you use the `repositoryInfo` tag to specify the Web Clipping repository settings.

Note: To determine the path of `provider.xml`, search for `webClipping/provider.xml` in the file system. For example, run the following command in UNIX:

```
> find DOMAIN_DIR -name "provider.xml" | grep -i webClipping
```

In Oracle JDeveloper's Integrated WLS, Web Clipping's `provider.xml` is located here:

JDEV_SYSTEM_DIRECTORY`/DefaultDomain/servers/DefaultServer/tmp/_WL_user/portalTools_11.1.1.2.0/`*RANDOMLY_GENERATED_DIRECTORY*`/war/WEB-INF/providers/webClipping/provider.xml`

Note that, on a Windows platform, pages in Framework applications are not rendered if there is a space in the path to the system directory in JDeveloper. Therefore, ensure that the *JDEV_SYSTEM_DIRECTORY* path does not contain spaces.

In the Fusion Middleware 11*g* installation, Web Clipping's `provider.xml` is located here:

FMW_HOME`/user_projects/domains/wc_domain/servers/WLS_Portlet/tmp/_WL_user/portalTools_11.1.1.2.0/`*RANDOMLY_GENERATED_DIRECTORY*`/war/WEB-INF/providers/webClipping/provider.xml`

This section includes the following subsections:

- Section E.2.1.1, "Using Oracle Metadata Services (MDS) as the Web Clipping Repository"
- Section E.2.1.2, "Using an Oracle Database as the Web Clipping Repository"

- Section E.2.1.3, "Configuring Web Clipping Repository in provider.xml"
- Section E.2.1.4, "Attributes and Child Tags of the repositoryInfo Tag"

E.2.1.1 Using Oracle Metadata Services (MDS) as the Web Clipping Repository

By default, in Oracle JDeveloper, the Web Clipping producer hosted on Integrated WLS, the default server, is configured to use MDS, which is file-based, as the Web Clipping repository.

> **Note:** In a full Oracle Fusion Middleware installation, the Web Clipping portlet producer is also included within the `WLS_Portlets` managed server in the default domain. By default, this Web Clipping portlet producer is configured to use the Oracle database, which is installed as part of Oracle WebLogic Server, as the Web Clipping repository.

Example E–1 shows MDS as the default repository in `provider.xml`.

Example E–1 MDS as the Default Web Clipping Repository in provider.xml

```
<repositoryInfo class="oracle.portal.wcs.provider.info.MdsInformation">
  <mdsConfigLocation>mds-config.xml</mdsConfigLocation>
</repositoryInfo>
```

For an MDS repository, the `repositoryInfo` tag is set to the `MdsInformation` class. The `mdsConfigLocation` entry specifies the path to the `mds-config.xml` file, which contains the metadata store configuration information, including the path to the actual metadata store. In Oracle JDeveloper, the `mds-config.xml` file is located at the following path:

JDEV_SYSTEM_DIRECTORY/DefaultDomain/servers/DefaultServer/tmp/_WL_user/portalTools_11.1.1.2.0/*RANDOMLY_GENERATED_DIRECTORY*/war/WEB-INF

> **Note:** On a Windows platform, pages in Framework applications are not rendered if there is a space in the path to the system directory in JDeveloper. Therefore, ensure that the *JDEV_SYSTEM_DIRECTORY* path does not contain spaces.

The `mds-config.xml` file specifies the location of the repository in a property tag:

```
<property name="metadata-path" value="portletdata/tools/webClipping"/>
```

The location specified for `value` is relative to *JDEV_HOME*`/portal`. Any relative path specified is interpreted to be relative to *JDEV_HOME*`/portal`. To use another location, for example, a location outside the Oracle JDeveloper home, then specify an absolute path, such as `c:\mds`.

> **Note:** For a multiple middle tier deployment, change the metadata path to a shared file system.

E.2.1.2 Using an Oracle Database as the Web Clipping Repository

Although MDS is the default repository in Oracle JDeveloper, you can alternatively select to use a database schema for your Web Clipping repository.

> **Note:** If you use a database as your Web Clipping repository, any customizations are retained even if the Web Clipping producer or the consuming application is redeployed.

You can use either of the following database schemas for Web Clipping:

- The default `PORTLET` database schema created by RCU for Oracle WebLogic Server
- A user-defined database schema for Oracle 9*i* or later

> **Note:** You cannot use a Microsoft SQL Server or IBM DB2 database as the Web Clipping repository.

This section includes the following subsections:

- Section E.2.1.2.1, "Using the Database Schema Created by RCU"
- Section E.2.1.2.2, "Creating a Database Schema"

E.2.1.2.1 Using the Database Schema Created by RCU For Web Clipping repository, you can use the Oracle database installed as part of Oracle WebLogic Server, through a JNDI data source named `jdbc/portletPrefs`. To access the database, the schema named `PORTLET` is used.

> **Note:** The `PORTLET` schema must be created in the Oracle database configured for WebCenter Portal. For more information, see the "Installing WebCenter Portal" chapter in the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

E.2.1.2.2 Creating a Database Schema You can use any user-defined schema for an Oracle 9*i* or later database as the Web Clipping repository. To create a database schema for Web Clipping definitions and clippings, run the Java command described in Example E–2.

Example E–2 Java Command for Creating a Schema for Web Clipping Portlet Definitions and Clippings

```
java -classpath WC_ORACLE_HOME/lib/xmlparserv2.jar:
WC_ORACLE_HOME/jdbc/lib/ojdbc14.jar:WC_ORACLE_HOME/portal/jlib/wce.jar
oracle.portal.wcs.Installer -installSchema -username dbuser -password
dbpassword -dburl jdbc:oracle:thin:@//host:port/dbid
```

Where:

- *WC_ORACLE_HOME* is the path to your WebCenter Portal Oracle home directory
- *dbuser* is the database user for the schema

  Consider using the same database user you use to create the WSRP and PDK-Java preference store database schema. If you do not use the same user, then you must create a new user and grant it connect and resource privileges.

- *dbpassword* is the specified user's password
- *dburl* is the URL for the database

  This is the database that contains the schema that you create for Web Clipping portlet definitions and clippings by using the following syntax:

  ```
  jdbc:oracle:thin:@//dbhost:dbport/service_name
  ```

  For example:

  ```
  jdbc:oracle:thin:@//shobeen:1521/sales_us
  ```

Note: The classpath in Example E–2 uses different separators for UNIX and Windows. On UNIX systems, the `classpath` uses a colon (:) separator. On Windows systems, the `classpath` uses a semicolon (;) separator.

E.2.1.3 Configuring Web Clipping Repository in provider.xml

To change the repository configuration for the Web Clipping producer deployed to Integrated WLS:

1. Open the `provider.xml` file in a text editor.
2. Specify the setting for your Web Clipping repository:
   - Use the `PORTLET` schema referred by the JNDI data source created by RCU. Specify the entries shown in Example E–3.

Example E–3 Oracle Database as the Web Clipping Repository

```
<repositoryInfo class="oracle.portal.wcs.provider.info.JdbcDbInformation">
   <connectionName>jdbc/portletPrefs</connectionName>
   <useRAA>false</useRAA>
   <useASO>false</useASO>
</repositoryInfo>
```

For information about tag parameters, see Section E.2.1.4, "Attributes and Child Tags of the repositoryInfo Tag."

   - Use the database schema, created for an Oracle database 9*i* or later, where you can manually specify connection information.

     To specify Oracle 9*i* or later database as the Web Clipping repository, specify database connection parameters in the entry shown in Example E–4.

Example E–4 Setting Oracle Database 9i or Later as Web Clipping Repository

```
<repositoryInfo class="oracle.portal.wcs.provider.info.DatabaseInformation">
   <useRAA>false</useRAA>
   <databaseHost>dbhost.mycompany.com</databaseHost>
   <databasePort>1521</databasePort>
   <databaseSid>iasdb</databaseSid>
   <databaseUsername>scott</databaseUsername>
   <databasePassword>!tiger</databasePassword>
   <useASO>false</useASO>
</repositoryInfo>
```

For information about tag parameters, see Section E.2.1.4, "Attributes and Child Tags of the repositoryInfo Tag."

If you require a secure database connection, then enable the Advanced Security Option (ASO) by setting the `useASO` entry to `true`. For more information about configuring ASO, see Section E.2.3, "Web Clipping Producer Security."

Note: Only one entry of `<repositoryInfo>` can exist in `provider.xml`. Depending on the Web Clipping repository you choose, you must uncomment only that entry.

---

3. Save the `provider.xml` file.
4. Restart Integrated WLS.

E.2.1.4 Attributes and Child Tags of the repositoryInfo Tag

Table E–6 lists and describes the attributes and child tags of the `repositoryInfo` tag.

Note: The attributes and child tags of the `repositoryInfo` tag are also described in the comments in the Web Clipping `provider.xml` file.

In the comments in the `provider.xml` file, the example provided for *Oracle9i Database or later using Oracle infrastructure database* is specific to Oracle Portal and its infrastructure database and application programming interfaces. That example must not be used for Framework application implementations.

Table E–6 Attributes and Child Tags of the repositoryInfo Tag

Attributes/Parameter	MDS/Database	Description
`class`	Both	The `class` attribute specifies the type of repository used to store Web Clipping definitions. The possible values for this attribute are: ■ `oracle.portal.wcs.provider.info.MdsInformation` This value signifies that MDS is used to store Web Clipping definitions and that MDS configuration is pushed to the `mds-config.xml` file. ■ `oracle.portal.wcs.provider.info.DatabaseInformation` This value signifies that an Oracle9*i* Database or later is used to store Web Clipping definitions and that the database connection details are included as children in the `repositoryInfo` tag. ■ `oracle.portal.wcs.provider.info.JdbcDbInformation` This value signifies that Oracle database installed as part of Oracle WebLogic Server is used to store Web Clipping definitions.
`mdsConfigLocation`	MDS	Use the `mdsConfigLocation` tag when the value for the `class` attribute indicates an MDS repository. It points to the MDS configuration file `mds-config.xml`, which specifies the actual MDS location. The `mds-config.xml` file is located at: *MW_HOME*`/wlserver_10.3/wc_domain/servers/WLS_Portlet/tmp/_WL_user/portalTools_11.1.1.2.0/yyggl7/war/WEB-INF`
`connectionName`	Database	Specifies the JNDI name of the data source where the Web Clipping repository has been installed using the RCU. By default, the connection name is `jdbc/portletPrefs`, which points to the `PORTLET` schema in Oracle WebLogic Server. In an interoperability scenario where an Oracle WebLogic Server 11*g* Middle Tier is paired with an Oracle Application Server 10*g* repository, the connection points to the `PORTAL` schema in the Oracle Application Server 10*g* repository.
`useASO`	Database	Set to `true` or `false`: ■ Specify `true` to use Oracle Advanced Security Option (ASO) to encrypt the communication channel between the Web Clipping producer and the database. This is provided for introducing added security if case-sensitive data is contained in the clipped content. ■ Specify `false` to omit this option.
`useRAA`	Database	Set to `true` or `false`: ■ Specify `true` if Repository Access APIs are used to access the database connection parameters. Specifying `true` is equivalent to making the Web Clipping producer use the default Oracle infrastructure database as the repository. Specifying `true` removes the need for other `repositoryInfo` child tags. ■ Specify `false` to omit this option.
`databaseHost`	Database	Specify the host name of the Oracle database. Use only version 9*i* or later. For example: `mycompany.dbhost.com`
`databasePort`	Database	Specify the port number of the Oracle database listener. This is usually `1521`.

Table E–6 (Cont.) Attributes and Child Tags of the repositoryInfo Tag

Attributes/Parameter	MDS/Database	Description
`databaseSid`	Database	Specify the Oracle SID of the database that hosts the Web Clipping repository.
`databaseUsername`	Database	Enter the user name used to log in to the database.
`databasePassword`	Database	Enter a plain text password for the specified database user name. Prefix the password with an exclamation point (!) to allow the Web Clipping producer to encrypt the password when the producer starts. For example: `!AX3tR`

E.2.2 HTTP or HTTPS Proxy Configuration

Your HTTP or HTTPS proxy settings must be set to allow the Web Clipping Studio to connect to web sites outside of your firewall. You can specify the settings by manually editing the `provider.xml` file.

Framework application administrators can set proxy settings manually according to the HTTP or HTTPS configuration. Edit the appropriate entries in the `provider.xml` file.

Example E–5 shows the relevant portion of `provider.xml`.

Example E–5 Proxy Settings

```
- <!--
 proxy information: Fill the following up if you have a proxy
 server between the provider and external sites.
   <proxyInfo class="oracle.portal.provider.v2.ProxyInformation">
      <httpProxyHost>proxy.mycompany.com</httpProxyHost>
      <httpProxyPort>80</httpProxyPort>
      <dontProxyFor>*.mycompany.com</dontProxyFor>
      <proxyUseAuth>true</proxyUseAuth>
      <proxyType>Basic</proxyType>
      <proxyRealm>realm1</proxyRealm>
      <proxyUseGlobal>false</proxyUseGlobal>
      <proxyUser>scott</proxyUser>
      <proxyPassword>!tiger</proxyPassword>
   </proxyInfo>

  -->
```

It is optional to specify values for the following tags: `<proxyUseAuth>`, `<proxyType>`, `<proxyRealm>`, `<proxyUseGlobal>`, `<proxyUser>`, and `<proxyPassword>`.

Table E–3, available earlier in this appendix, describes the proxy settings you specify in the `provider.xml` file. The descriptions in the table are applicable for Web Clipping producers also.

Note: For environments that use a proxy server to reach external web sites, you can use the `dontProxyFor` entry to specify the proxy exception list. Web Clipping uses the proxy exception list to restrict users from clipping content from unauthorized external web sites. Users attempting to reach a web site in a listed domain, from the Web Clipping Studio, receive an HTTP timeout error.

E.2.3 Web Clipping Producer Security

The preceding sections described the administrative tasks that must be performed before you are able to use the Web Clipping producer. The following sections describe some security configuration options that you must implement to enable the Web Clipping producer to access Secure Sockets Layer (SSL)-enabled web sites and encrypt the channel between itself and the database:

- Section E.2.3.1, "Adding Certificates for Trusted Sites"
- Section E.2.3.2, "Configuring Oracle Advanced Security for the Web Clipping Producer"

E.2.3.1 Adding Certificates for Trusted Sites

When a user navigates to a secure site, the web site typically returns a certificate, identifying itself to the user when asking for secure information. If the user accepts the certificate, then the certificate is placed into the list of trusted certificates of the browser so that a secure channel can be opened between the browser and that server. Like a web browser, the Web Clipping producer acts as an HTTP client to external web sites. To keep track of trusted sites, the Web Clipping producer uses the `cacerts` file, which is the Java keystore to store trusted certificates.

By default, the `cacerts` file stores various certificates including the Equifax, VeriSign, and Cybertrust certificates. However, it does not include all possible server certificates that exist on the web. For this reason, when a user navigates to a secure server using HTTPS, the user can get an SSL handshake failed exception in the Web Clipping Studio. To solve this problem, you must add the certificate of that site to the `cacerts` file.

To add a certificate to the `cacerts` file:

1. Download the certificate of the HTTPS web site and save the certificate in the PEM format.

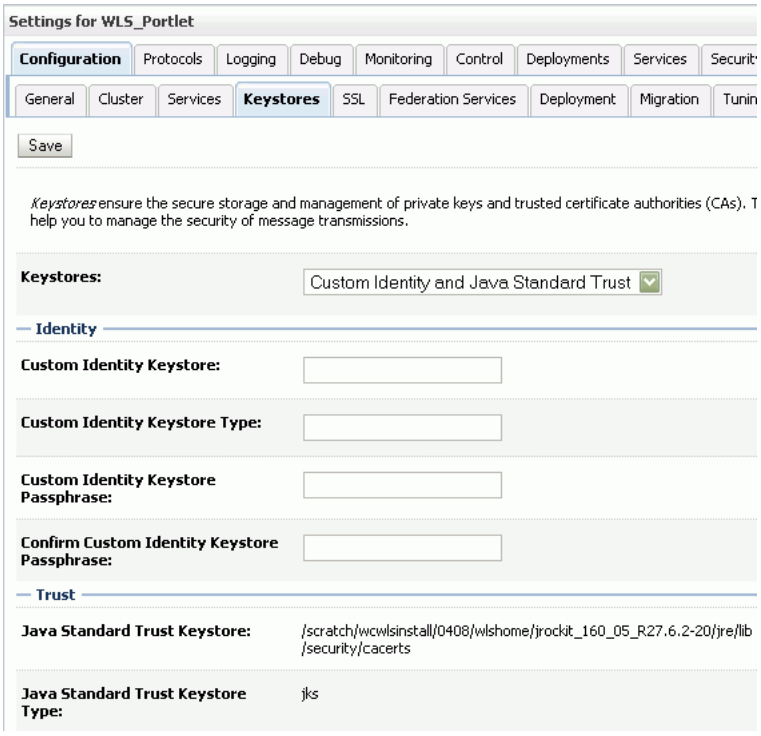
   Tip: Mozilla Firefox 3.0 or later enable you to save a certificate file in the PEM format.

2. Locate the path to the `cacerts` file:
   a. Log in to the Oracle WebLogic Server Administration Console by using the following URL format:

      `http://host:port/console`

   b. Open the **Keystore** tab for the **WLS_Portlets** managed server.
   c. Note down the location of the `cacerts` file given in the **Java Standard Trust Keystore** field.

Figure E–2 Locating the cacerts file



3. At the command prompt, navigate to the location of the `cacerts` file and run the following command to add the certificate:

   `keytool -importcert -alias` *`certi_alias`* `-file` *`certifi_name`* `-keystore cacerts -storepass` *`password`*

   Where, *`certi_alias`* refers to the alias used for the certificate, *`certifi_name`* refers to the certificate file name, and *`password`* refers to the password of the `cacerts` file. The default password is `changeit`.

   For example,

   ```
   keytool -importcert -alias stamf05 -file stamf05.crt
   -keystore cacerts -storepass changeit
   ```

   Tip: It is advisable to import trusted certificate into the `cacerts` file by using an alias. It makes it easier for you to delete or list the entries in the keystore.

E.2.3.2 Configuring Oracle Advanced Security for the Web Clipping Producer

The Web Clipping producer can use Oracle Advanced Security Option (ASO) to secure and encrypt the channel between itself and the database that hosts the Web Clipping repository. This feature is available only if you have selected any Oracle database as the Web Clipping repository. This feature is disabled by default as Oracle Metadata Services is the default Web Clipping repository in Oracle JDeveloper.

To enable ASO:

1. Open `provider.xml` in a text editor.
2. Under the repository settings section in the file (shown in Example E–3), set the `useASO` entry to `true`.
3. Save the `provider.xml` file.

In addition, you must set the following ASO configuration parameters in the `sqlnet.ora` file to ensure that the database connections created between the Web Clipping producer and the database hosting the Web Clipping repository are using ASO.

- `SQLNET.AUTHENTICATION_SERVICES` -- This parameter is used to select a supported authentication method in making database connections with ASO. For more information about setting this parameter, see the *Oracle Database Advanced Security Administrator's Guide*.
- `SQLNET.CRYPTO_SEED` -- This parameter denotes the cryptographic seed value (FIPS 140-1 setting), used in making database connections with ASO.

  For more information about setting parameters and about the values to use for various possible combinations of parameters, see the *Oracle Database Advanced Security Administrator's Guide*.

> **Note:** When setting these parameters after the initial configuration (where the database parameters are already set up), database connections are assumed to be open. Because enabling the ASO affects all connections made to the database, it is advisable to restart Integrated WLS containing the Web Clipping producer to reset all the current connections to use ASO. You must also do this when disabling ASO.

E.3 Hiding or Removing the WSRP Test Page

If you do not want all users to be able to see the WSRP test page, you can protect it so that only administrators can see it. To do this, add the code shown in Example E–6 to the `web.xml` file.

Example E–6 Hiding the WSRP Test Page

```
<security-role>
  <description>AdministratorRole</description>
  <role-name>Admin</role-name>
</security-role>
<security-constraint>
  <display-name>TestPageInfo</display-name>
  <web-resource-collection>
    <web-resource-name>TestPageInfo</web-resource-name>
    <description>Protect the test page servlet.</description>
    <url-pattern>/info/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <description>Administrators</description>
    <role-name>Admin</role-name>
  </auth-constraint>
  <user-data-constraint>
    <transport-guarantee>NONE</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

If you want to remove the WSRP test page completely, you must modify the `web.xml` file to comment out the element shown in Example E–7.

Note: This element is injected into `web.xml` at packaging time, so you must edit the `web.ml` file in the resulting EAR file.

Example E–7 Removing the WSRP Test Page

```
<servlet-mapping>
   <servlet-name>WSRPTestPage</servlet-name>
   <url-pattern>/info</url-pattern>
</servlet-mapping>
```

E.4 Setting Up a Preference Store

The portlet preference store is used for persisting consumer registration handles and portlet preference data. Portlet producers can use one of three types of preference store: *File*, *Database* and *Consumer*.

Using a consumer preference store ensures that the preference store data is always stored with the consumer application. Whatever happens to the producer over time, the consumer application's registrations and portlets continue to be valid while the producer connections point to functionally compatible producers (that is, the producer should be able to decode the portlet name from the portlet handle and be able to resolve this to the portlet that the consumer application developer expects). When this mechanism is used, the producer itself stores no persistent state whatsoever, meaning that a producer can be switched for another compatible one at any time, with no requirement to perform a preference store migration or export/import cycle.

In a clustered environment, Oracle recommends the use of a database or consumer preference store. If you prefer to use a file-based preference store, then you must also use a shared file system.

If you configure the WebLogic Server to run Portlet Producer applications, then the database preference store is configured for you. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

This section contains configuration information for the following areas:

- Section E.4.1, "WSRP Producers"
- Section E.4.2, "PDK-Java Producers"

E.4.1 WSRP Producers

WSRP producers have a JNDI preference value that determines which preference store is used, and is set in the `web.xml` file of the Portlet Producer application. Table E–7 lists and describes the JNDI variables used to specify the preference store for WSRP producers.

Table E–7 WSRP Producer Database Preference Store-Related JNDI Variable

Variable Name	Variable Value	Description

`oracle/portal/wsrp/server/persistentStore`	`Database`
	`File`
	`Consumer`
		Determines which data store (File, Database, or Consumer) is used for persisting a Portlet Producer application's consumer registration handles and portlet preferences.

`oracle/portal/wsrp/server/fileStoreRoot`	`portletdata`	Defines the path to the root directory to be used by the file preference store. Absolute paths are interpreted relative to the file system root. Relative paths are interpreted relative to the *`WC_ORACLE_HOME`*`/portal` directory. Note that all producers running within the same WebLogic Server must use the same path for this variable. Otherwise, you get a `Portlet unavailable` error for some portlets.

The following examples show the entries in a producer's `web.xml` file.

Example E–8 Configuring web.xml to Use a Database Preference Store

```
<env-entry>
   <env-entry-name>oracle/portal/wsrp/server/persistentStore</env-entry-name>
   <env-entry-type>java.lang.String</env-entry-type>
   <env-entry-value>Database</env-entry-value>
</env-entry>
```

Example E–9 Configuring web.xml to Use a File-Based Preference Store

```
<env-entry>
  <env-entry-name>oracle/portal/wsrp/server/persistentStore</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>File</env-entry-value>
</env-entry>
<env-entry>
  <env-entry-name>oracle/portal/wsrp/server/fileStoreRoot</env-entry-name>
  <env-entry-type>java.lang.String</env-entry-type>
  <env-entry-value>myPrefStore</env-entry-value>
</env-entry>
```

Example E–10 Configuring web.xml to Use a Consumer Preference Store

```
<env-entry>
   <env-entry-name>oracle/portal/wsrp/server/persistentStore</env-entry-name>
   <env-entry-type>java.lang.String</env-entry-type>
   <env-entry-value>Consumer</env-entry-value>
</env-entry>
```

When you have updated the relevant `web.xml` files, restart your WebLogic Server.

E.4.2 PDK-Java Producers

The type of preference store used for PDK-Java producers is determined by the `preferenceStore` tag in the `provider.xml` file. Table E–8 lists and describes the attributes and parameters used with the `preferenceStore` tag when a database preference store is specified.

Table E–8 Attributes and Parameters of the preferenceStore Tag

Attribute/Parameter	Description

`class`	This required attribute specifies the Java class that defines the location and other details of portlet preferences. For example, a file-based preference store might use:
	`<preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore ">`
	A database preference store might use:
	`<preferenceStore class="oracle.portal.provider.v2.preference.DBPreferenceStore">`
	OmniPortlet uses its own class, for example:
	`<preferenceStore class="oracle.webdb.reformlet.ReformletFilePreferenceStore">`

`name`	This required parameter provides a name for the preference store. Use any value you choose. For example:
	`<preferenceStore class="oracle.portal.provider.v2.preference.DBPreferenceStore">`
	`<name>MyPDKProducerPreferenceStore</name>`
	`</preferenceStore>`

`connection`	This required parameter for a database preference store points to a JNDI connection that connects with a schema containing the portlet preference store. For example:
	`<preferenceStore class="oracle.portal.provider.v2.preference.FCFDBPreferenceStore">`
	`<name>MyPDKProducerPreferenceStore</name>`
	`<connection>java:comp/env/jdbc/portletPrefs</connection>`
	`</preferenceStore>`

`rootDirectory`	This optional parameter specifies the location where the file-based preference store preferences are stored. When this parameter is not included with the `preferenceStore` tag, the default file-based preference store location is used. This is the same folder where the producer's provider definition file is stored.

`useHashing`	This optional parameter is used when specifying a file based preference store, and accepts the value `true` or `false`. When it is set to `true`, each preference data file is stored in an extra subdirectory with a name determined by hashing the data file name. Using this parameter can improve file system performance by limiting the number of preference data files stored in a single directory.
	For example:
	`<preferenceStore class="oracle.portal.provider.v2.preference.FilePreferenceStore">`
	`<name>PDKProducerPreferenceStore</name>`
	`<useHashing>true</useHashing>`
	`</preferenceStore>`

E.5 Portlet Preference Store Migration Utilities

A preference store is a mechanism for storing information like user preference data, portlet and producer settings, or even portlet data. Preferences can be stored in a database, which is recommended for high availability configurations, a file system, or with the consuming application. You can migrate the following stores for your Framework applications:

- Section E.5.1, "JPS Portlet Preference Store - PersistenceMigrationTool"
- Section E.5.2, "PDK-Java Portlet Preference Store - Migration and Upgrade Utilities"
- Section E.5.3, "Web Clipping Repository"
- Section E.5.4, "Moving a Producer"

E.5.1 JPS Portlet Preference Store - PersistenceMigrationTool

A WSRP container preference store is a mechanism used for persisting consumer registration and portlet preference data. Currently, there are three preference store implementations, *database preference store*, *file-based preference store*, and *consumer preference store*. A database preference store persists data using a relational database. A file-based preference store persists data using the file system. A consumer preference store ties the producer metadata to the consumer application. A file-based or consumer preference store may be used for testing, since it removes the dependency on a database. But, for highly available configurations, Oracle recommends that you use a database or consumer preference store.

The WSRP container preference store migration utility, `PersistenceMigrationTool`, enables you to migrate existing data between database and file-based preference stores (for example, from a database preference store to a file preference store). This utility also enables upgrading users to ensure that their existing locale-specific portlet preference data uses a naming format compatible with the latest JPS release. You can also use this utility to migrate between source and destination stores of the same type, enabling data to be moved from one database store to another.

Note: You cannot use the preference store migration utility to migrate to or from a consumer preference store. To migrate to or from a consumer preference store, you must export the data from one producer from the consumer and import into another.

1. Export the producer metadata from your consumer application to an EAR file, using JDeveloper or WLST. This contacts the remote producer to get customization data, and so on.
2. Either remap the producer connection to another producer, which can be using any preference store type, or change the preference store configuration of the current producer, and restart it.
3. Import from the exported EAR file back to your consumer application. This pushes the relevant metadata back to the producers, which then store it in their configured preference store.

For information about exporting and importing producers using JDeveloper, see Section E.6, "Exporting and Importing Portlet Producers at Design Time." For information about exporting and importing producers using WLST, see the section "Exporting and Importing Custom WebCenter Applications for Data Migration" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

The syntax of the `PersistenceMigrationTool` is:

```
java oracle.portlet.server.containerimpl.PersistenceMigrationTool
-sourceType [file | db]
-destType [file | db]
{-sourcePath [dir |
 -sourceUsername username -sourcePassword password -sourceDatabase db]}
{-destPath [dir | destUsername username -destPassword password -destDatabase db]}
[-debug]
```

where:

`sourceType` indicates whether the source store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

`destType` indicates whether the destination store is in a file or database. You can have source and destination stores of the same type. Hence, you can migrate from one database to another or one file system to another.

`sourcePath` is the location of a file-based preference store. This argument is required when `sourceType` is `file`.

`sourceUsername` is the database user name for a preference store database. This argument is required when `sourceType` is `db`.

`sourcePassword` is the database password for a preference store database. This argument is required when `sourceType` is `db`.

`sourceDatabase` is the name of a preference store database. This argument is required when `sourceType` is `db`.

`destPath` is the location of a file-based preference store. This argument is required when `destType` is `file`.

`destUsername` is the database user name for a preference store database. This argument is required when `destType` is `db`.

`destPassword` is the database password for a preference store database. This argument is required when `destType` is `db`.

`destDatabase` is the name of a preference store database. This argument is required when `destType` is `db`.

`debug` turns on full logging through standard output to enable users to diagnose issues that arise when the tool runs.

Note: There are several libraries that must be referenced in the classpath when running the `PersistenceMigrationTool` utility:

- `wcs-producer-spi.jar`
- `portlet-utils.jar`
- `portlet-producer-container-common.jar`
- `portlet-producer-container-persistence.jar`
- `oracle-portlet-api.jar`
- `wsrp-container.jar`
- `oracle-portlet-tags.jar`
- `ojdbc6.jar`

The `ojdbc6.jar` library referenced in the classpath must be the same as the one used by your database.

Example E–11 demonstrates running the `PersistenceMigrationTool` utility. In this example, preferences from a database store are copied to a file store.

Example E–11 Running the PersistenceMigrationTool Utility

```
./java  -classpath
ORACLE_COMMON_HOME/webcenter/modules/com.bea.wsrp_10.3.2.0/system/wcs-producer-spi.jar:
ORACLE_COMMON_HOME/webcenter/modules/oracle.portlet.server_11.1.1/portlet-utils.jar:
ORACLE_COMMON_HOME/webcenter/modules/oracle.webcenter.framework_
11.1.1/portlet-producer-container-common.jar:
ORACLE_COMMON_HOME/webcenter/modules/oracle.webcenter.framework_
11.1.1/portlet-producer-container-persistence.jar:
WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/oracle-portlet-api.jar:
WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/wsrp-container.jar:
WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/oracle-portlet-tags.jar:
DB_ORACLE_HOME/jdbc/lib/ojdbc6.jar
oracle.portlet.server.containerimpl.PersistenceMigrationTool
-sourceType db \
-sourceUsername scott \
-sourcePassword tiger \
-sourceDatabase abc.mycompany.com:1521:yourdatabase \
-destType file \
-destRoot /data/prefs
```

where:

- *ORACLE_COMMON_HOME* is your Oracle Common home
- *WC_ORACLE_HOME* is your WebCenter Portal Oracle home
- *DB_ORACLE_HOME* is your database Oracle home if it is on the same machine as the WebCenter Portal Oracle home. If the database Oracle home is on a separate

machine, then you must copy the *DB_ORACLE_HOME/jdbc/lib* directory into a temporary directory on the WebCenter Portal Oracle home and reference the `ojdbc6.jar` library from this temporary directory in the classpath.

E.5.2 PDK-Java Portlet Preference Store - Migration and Upgrade Utilities

PDK-Java has two `PreferenceStore` implementations, `DBPreferenceStore` and `FilePreferenceStore`. `DBPreferenceStore` persists data using a JDBC-compatible relational database and `FilePreferenceStore` persists data using the file system.

If you have installed Oracle PDK, then you can manage the information stored in the preference store by using the Preference Store Migration and Upgrade Utility, which is included in the `pdkjava.jar` file. Note that you must run this tool from `WC_ORACLE_HOME`. The syntax of the migration utility is:

Note: You must set the classpath when running this tool to include `pdkjava.jar`, `ptlshare.jar`, and `ojdbc6.jar`. For example:

```
java -classpath WC_ORACLE_HOME\jdeveloper\modules\oracle.dms_
11.1.1\dms.jar;
WC_ORACLE_HOME\wlserver_10.3\server\lib\ojdbc6.jar;WC_ORACLE_
HOME\jdeveloper\webcenter\modules\oracle.portlet.server_
11.1.1\pdkjava.jar;
WC_ORACLE_HOME\jdeveloper\webcenter\modules\oracle.portlet.server_
11.1.1\ptlshare.jar
oracle.portal.provider.v2.preference.MigrationTool
```

```
java oracle.portal.provider.v2.preference.MigrationTool
  -mode [file | db | filetodb | filetofile | dbtofile | dbtodb]
  [-remap language | locale]
  [-countries iso_country_code]
  [-pref1UseHashing true | false]
  [-pref1Driver driver]
  {-pref1RootDirectory directory |
   -pref1User username -pref1Password password -pref1URL url}
  [-pref2UseHashing true | false]
  [-pref2Driver driver]
  {-pref2RootDirectory directory |
   -pref2User username -pref2Password password -pref2URL url}
  [-upfixwpi filename]
```

where

- `-mode` is the mode in which you want to run the Preference Store Migration and Upgrade Utility

  `filetodb`, `filetofile`, `dbtofile`, or `dbtodb` indicates to run in migration mode. See Section E.5.2.1, "Migration Mode" for more information about this mode.

  `file` or `db` indicates to run in upgrade mode. See Section E.5.2.2, "Upgrade Mode" for more information about this mode.

- `-remap` is the `localePersonalizationLevel` (language or locale). Note that you only must use this option to change `localePersonalizationLevel` as part of your upgrade or migration.

- `-countries` specifies a prioritized list of ISO country codes, indicating your order of preference in a collision between remapped preferences for different countries. `-countries` is only meaningful if you also specified the `-remap` option.
- `-pref1UseHashing` specifies whether you want to employ hashing on the source for this operation.
- `-pref1Driver` is the driver for the source database. If you do not specify this parameter, the nearest matched driver is used.
- `-pref1RootDirectory` is the path of a source file system, for example, `j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample`.
- `-pref1User` is the user name for a source database.
- `-pref1Password` is the password for a source database.
- `-pref1URL` is the URL to a source database, for example, `jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid`.
- `-pref2UseHashing` specifies whether you want to employ hashing on the destination for this operation.
- `-pref2Driver` is the driver for the destination database. If you do not specify this parameter, the nearest matched driver is used.
- `-pref2RootDirectory` is the path of a destination file system, for example, `j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample`.
- `-pref2User` is the user name for a destination database.
- `-pref2Password` is the password for a destination database.
- `-pref2URL` is the URL to a destination database, for example, `jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid`.
- `-upfixwpi` indicates a log file for the operation.

This section includes the following subsections:

- Section E.5.2.1, "Migration Mode"
- Section E.5.2.2, "Upgrade Mode"

E.5.2.1 Migration Mode

Use a migration mode to copy data from a source preference store to a target preference store. When the utility is run in this mode, the preference stores for all the portlet definitions are updated.

Table E–9 describes the migration modes in which you can run the utility.

Table E–9 Migration Modes in Which to Run the Utility

Mode	Description
`filetodb`	Use when data must be copied from a `FilePreferenceStore` to a `DBPreferenceStore`.
`filetofile`	Use when data must be copied from one `FilePreferenceStore` to another `FilePreferenceStore` that is in a different location.
`dbtofile`	Use when data must be copied from a `DBPreferenceStore` to a `FilePreferenceStore`.

Table E–9 (Cont.) Migration Modes in Which to Run the Utility

Mode	Description
dbtodb	Use when data must be copied from one DBPreferenceStore to another DBPreferenceStore that is based on a different database table.

If the destination for the operation is a database, you must ensure that the destination WebLogic Server is created using the WebCenter Portal portlet template and the appropriate schemas have been created using the RCU. For more information, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

When using a migration mode, you can also use the -remap and -countries options to specify that the data must be upgraded while being migrated. Specifically, you use these options to ensure that the locale-specific preferences are appropriately remapped.

You can use the other options accepted by the utility to specify the properties of the preference stores involved in the upgrade or migration process. These options must correspond to the tags you specify in provider.xml to describe the preference stores.

Properties beginning with the prefix -pref1 correspond to properties of the source preference store (in an upgrade mode, this is the only preference store). For example, specifying -pref1UseHashing true -pref1RootDirectory j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample sets the useHashing and rootDirectory properties of a source FilePreferenceStore.

When a migration basic mode is selected, properties beginning with the prefix -pref2 correspond to properties of the target preference store. For example, specifying -pref2User portlet_prefs -pref2Password portlet_prefs -pref2URL jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid sets the database connection details on a target DBPreferenceStore.

Example E–12 PDK-Java Migration Utility Command Line, Migration

```
java -classpath WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_
11.1.1/pdkjava.jar:
WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/ptlshare.jar:WC_
ORACLE_HOME/ucm/shared/classes/ojdbc14.jar \
oracle.portal.provider.v2.preference.MigrationTool \
-mode dbtofile \
-pref1User portlet_prefs \
-pref1Password portlet_prefs \
-pref1URL jdbc:oracle:thin:@myserver.mydomain.com:1521:mysid \
-pref2RootDirectory /mydirectory/preferences
```

E.5.2.2 Upgrade Mode

Use an upgrade mode to upgrade data in place, and to modify existing locale-specific preferences in the preference store so that the naming format used is compatible with the current version of Oracle Portal and a given localePersonalizationLevel setting.

Table E–10 describes the upgrade modes in which you can run the utility.

Table E–10 Upgrade Modes in Which to Run the Utility

Mode	Description
`file`	Use when data in a `FilePreferenceStore` must be upgraded.
`db`	Use when data in a `DBPreferenceStore` must be upgraded.

An upgrade mode can be used in the following scenarios:

- You have upgraded from Oracle PDK 9.0.4.0.0 or earlier and want to use existing portlets with the default `localePersonalizationLevel` setting of language (In earlier releases, the default setting was `locale`).
- You have upgraded from Oracle Portal 9.0.2.0.0 or earlier and want to use existing portlets with a `localePersonalizationLevel` setting of `locale` (Oracle Portal now uses different names for some locales and therefore some existing data must be remapped).
- You want to change the `localePersonalizationLevel` for an existing portlet from `locale` to `language` or vice-versa.

When using an upgrade mode, you must use the `-remap` option to specify the `localePersonalizationLevel` (`language` or `locale`) that you are upgrading to. You can also use the `-countries` option to specify a prioritized list of ISO country codes, indicating your order of preference in a collision between remapped preferences for different countries. For example, if you specify `-remap language -countries GB,US` in the command, then it means that if the utility comes across both US English and British English preferences (`en_US` and `en_GB`) in a given preference store, it remaps the British English preference to become the English-wide preference (`en`).

Example E–13 PDK-Java Migration Utility Command Line, Upgrade

```
java -classpath WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_
11.1.1/pdkjava.jar:
WC_ORACLE_HOME/webcenter/modules/oracle.portlet.server_11.1.1/ptlshare.jar:WC_
ORACLE_HOME/ucm/shared/classes/ojdbc14.jar \
  oracle.portal.provider.v2.preference.MigrationTool \
 -mode file -remap language
 -countries GB,US -pref1UseHashing true
 -pref1RootDirectory j2ee/home/applications/jpdk/jpdk/WEB-INF/providers/sample
```

E.5.3 Web Clipping Repository

Web Clipping does not have a preference store as such, but it stores Web Clipping definitions and associated metadata. By default, it uses MDS, which is file-based, for this purpose. But you can configure Web Clipping to use a database. To migrate this repository for Framework applications, you can use the Predeployment tool in export and import mode to go from MDS to a database, or vice versa. This procedure must be done for each application as follows:

1. Run the Predeployment tool in `export` mode on all Framework applications that use the Web Clipping producer. For more information, see Section 61.2.10, "How to Implement Export of Customizations (WSRP 2.0)."
2. Update the producer to use a different repository. For information, see Section E.2.1.3, "Configuring Web Clipping Repository in provider.xml."

3. Run the Predeployment tool in `import` mode on all Framework applications that use the Web Clipping producer. For more information, see Section 61.2.10, "How to Implement Export of Customizations (WSRP 2.0)."

E.5.4 Moving a Producer

In some situations, you may want to move a portlet producer. To do so, you must:

1. Install the new producer.
2. Make the preference store of the original producer available to the new producer by performing one of the following tasks:
   - Migrate the preference store using the Preference Store Migration and Upgrade Utilities (see Section E.5.2, "PDK-Java Portlet Preference Store - Migration and Upgrade Utilities" for more information), or
   - Configure the new producer to use the same preference store as the original producer. If the database preference store is being used, then you must point the WebLogic Server data source to the same database as the original producer.
3. Update the URL of the producer registration by using Enterprise Manager Fusion Middleware Control or the WLST commands: `setWSRPProducerRegistration` or `setPDKJavaProducerRegistration`.

E.6 Exporting and Importing Portlet Producers at Design Time

You can export and import portlet producers at design time.

This section includes the following subsections:

- Section E.6.1, "Exporting Portlet Producers at Design Time"
- Section E.6.2, "Importing Portlet Producers at Design Time"

E.6.1 Exporting Portlet Producers at Design Time

When you package an application for deployment, any portlet producers referenced by the application are contacted so that the producer data can be included in the MAR file. If any of the producers are not running, they cannot be contacted and the data cannot be included.

By creating an export archive of the producer data at design time instead, you can ensure that the producers are running, therefore all the producer data, including remote customizations and client-side metadata, can be retrieved. WebCenter Portal: Framework can then use the export archive at deployment, instead of having to contact the remote producers.

To export portlet producers at design time:

1. Edit the *`WC_ORACLE_HOME`*`/jdeveloper/jdev/bin/jdev.conf` file and set the following JVM flag:

   ```
   AddVMOption  -Doracle.webcenter.portlet.dt.disableRemoteExport=true
   ```

   Setting this flag ensures that, when the application is packaged for deployment, the export archive you create in the following steps is included in the MAR file, rather than the remote producers being contacted to retrieve producer data.
2. Restart JDeveloper to apply the new setting.

3. Open the application that consumes the producers to export.
4. From the menu, choose **Application** and then **Export Portlet Producers**.

   This menu option appears only if the current application contains producers.
5. In the Export Portlet Producers dialog, in the **Export Archive File Name (.ear)** field, enter the absolute path and file name to use for the export set.
6. Click **OK**.

   When the application is packaged for deployment, because you set the `disableRemoteExport` flag to `true`, WebCenter Portal: Framework checks for the presence of the export archive at the location specified in the dialog. If the export archive exists, the contents of the archive are included in the MAR file instead of contacting the remote producers to retrieve producer data.

   Note: If the `disableRemoteExport` flag is set and there is no export archive, a default export archive, without the remote producer data, is created and included in the MAR file.

E.6.2 Importing Portlet Producers at Design Time

You can import producer data, including customizations made at runtime, from a deployed application into your application at design time.

Note: The producers in the export archive (EAR file) must be the same as those in the application into which they are being imported.

For information about how to create an export archive of producer metadata from a deployed application, see the section "Exporting Portlet Client Metadata (WebCenter Portal: Framework Application)" in the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

To import portlet producers at design time:

1. Edit the *`WC_ORACLE_HOME`*`/jdeveloper/jdev/bin/jdev.conf` file and set the following JVM flag:

   ```
   AddVMOption  -Doracle.webcenter.portlet.dt.enableImport=true
   ```
2. Restart JDeveloper to apply the new setting. This ensures that the **Import Portlet Producers** menu option is available.
3. Open the application into which you want to import the producers.
4. From the menu, choose **Application** and then **Import Portlet Producers**.
5. In the Import Portlet Producers dialog, in the **Import Archive File Name (.ear)** field, enter the absolute path and file name of the export archive to import.
6. Click **OK**.

   Note: If the application into which you import producers is currently running in Integrated WLS, you must re-run the application to see the updated producers. Simply refreshing the page may result in errors caused by the different MDS instances used at design time and runtime.

E.7 Migrating WebLogic Portal Portlets to WebCenter Portal

This section discusses migrating WebLogic Portal portlets developed with the WebLogic Portal's Eclipse IDE to WebCenter Portal's JDeveloper environment.

Note: There is no direct support or tooling available for migrating WLP portlets to a WebCenter Portal project. In many cases, significant refactoring and recoding is required. This section highlights some of the migration issues and offers recommendations as appropriate.

E.7.1 Migrating Java Portlets from WebLogic Portal to WebCenter Portal

In general, JSR 286 standard portlets (Java portlets) that were developed in WebLogic Portal can be moved directly to a WebCenter Portal/JDeveloper environment. Simply copy all of the portlet artifacts (`portlet.xml`, `.java` files, `.jsp` files, and so on) into a JDeveloper Portlet Producer application project.

Note: Any WebLogic Portal specific APIs used by the portlet must be rewritten to use WebCenter Portal APIs. WebLogic Portal specific APIs will not work in a WebCenter environment.

Tip: You can use the WebLogic Portal Export feature to export your Java portlets to an archive file and then import the archive file into your JDeveloper project. This technique may be simpler than manually copying all the portlet artifacts from one environment to another. See "Exporting Java Portlets for Use on Other Systems" in the *Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal*.

E.7.2 Problems With Migrating WLP Portlets to a WebCenter Portal Application

Moving portlets from a WLP development environment to a WebCenter Portal development environment is not directly supported. In general, this process can involve substantial rewriting or refactoring of the migrated portlet code and related files.

Note: If a portlet (regardless of type) was capable of running over WSRP in WebLogic Portal, the portlet can be consumed directly from the WebLogic Portal's portlet producer in a WebCenter Portal consumer. See "WSRP Interoperability with WebCenter Framework" in the *Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal*.

Problems inherent in moving WebLogic Portal portlets directly to a WebCenter Portal project in JDeveloper can include the following:

- **URL Generation** – Some URL types supported by WebLogic Portal do not work in WebCenter Portal, including DesktopURL, CustomEventURL, PageURL, WindowURL, StandalonePortletURL, and possibly others.
- **Events** – The WebCenter Portal consumer does not generate all of the events that the WebLogic Portal framework generates. These unsupported events include Init,

LookAndFeelReinit, Notification, Refresh, WindowActivation, WindowDeactivation, and possibly others.

- **Render Dependencies** – WLP render dependencies do not work in WebCenter.
- **WLP Framework APIs** – Many WLP APIs are not supported in WebCenter Portal.

E.7.3 General Tips for Migrating WLP Portlets to a WebCenter Portal Portlet Producer Application

This section provides general guidance on moving WLP portlets into WebCenter Portal's portlet producer environment. Although portlet migration is not directly supported, this section lists tips that might help you with the manual migration process of the various WLP portlet types.

> **Caution:** This section provides general guidance only. As discussed in Section E.7, "Migrating WebLogic Portal Portlets to WebCenter Portal," direct migration of WLP portlet types to WebCenter Portal application or portlet producer application is not directly supported. In almost all cases, rewriting and refactoring of existing portlet code is required when moving portlets from WLP to WebCenter Portal.
>
> As noted previously, any WebLogic Portal specific APIs used by the portlet must be rewritten to use WebCenter Portal APIs. WebLogic Portal specific APIs will not work in a WebCenter environment.

- **JSP portlets** – Moving JSP portlets directly into a WebCenter Portal project is not supported. You can consider refactoring the JSP portlet into a JSR286 portlet and then migrate it as explained in Section E.7.1, "Migrating Java Portlets from WebLogic Portal to WebCenter Portal."
- **JSR 168/286 portlets** – Most Java (JSR 168 / 286) portlets can be directly imported to a WebCenter Portal portlet producer and run as JSR286 portlets. Some JSR168 portlets that take advantage of specific error conditions guaranteed by the JSR168 specification may need to be run in a JSR168 compatibility mode. See "JSR-286/JSR-168 Portlet Compatibility" in the *Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal*. JSR168 portlets using WLP's proprietary eventing (event subscriptions declared in a `.portlet` file) must be re-written to use JSR286 events.
- **Java Page Flow portlets** – JPF portlets are not supported by WebCenter Portal's portlet producer and must be either consumed from a WebLogic Portal WSRP producer or refactored to become JSR286 or JSF portlets.
- **JSF portlets** – If the portlet is written to the JSR329 JSF Portlet bridge in WLP, it should run on a WebCenter producer with no changes. For portlets using the WLP "native" JSF portlet bridge, the portlet must be consumed from a WebLogic Portal producer or upgraded to a JSF 1.2 portlet using the JSR329 bridge. See also "Working With JSF-Java Portlets" in the *Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal*.
- **Clipper portlets** – Clipper portlets are not supported in WebCenter portal, but the web clipping features in WebCenter Portal's pagelet producer provide equivalent functionality.
- **Struts portlets** – Moving Struts portlets directly into a WebCenter Portal project is not supported. You can consider refactoring the JSP portlet into a JSR286 portlet

and then migrate it as explained in Section E.7.1, "Migrating Java Portlets from WebLogic Portal to WebCenter Portal."

- **Content Presenter portlets** – WLP Content Presenter portlets will not work over WSRP and will not work with WebCenter. Portal However, equivalent functionality is available in WebCenter Portal's content presenter. See "Publishing Content Using Content Presenter" in the *Oracle Fusion Middleware User's Guide for Oracle WebCenter Portal: Spaces*.
- **Remote (WSRP) portlets** – Remote (WSRP) portlets consumed in WLP can be consumed in a WebCenter Portal's consumer instead. Remote portlets taking advantage of WLP-specific WSRP features may need modification. For example the custom data transfer feature must be replaced by using events or shared parameters to convey data. See "WSRP Interoperability With WebCenter Framework" and "Configuring WSRP Security Between WLP and WebCenter Framework" in the *Oracle Fusion Middleware Federated Portals Guide for Oracle WebLogic Portal*.

F

Reuse of Oracle Portal Components

This appendix describes how to reuse components from your existing Oracle Portal application in an application built with Oracle WebCenter Portal: Framework.

This appendix includes the following sections:

- Section F.1, "Introduction to Oracle Portal Components"
- Section F.2, "Reusing Portlets"
- Section F.3, "Reusing Items"

F.1 Introduction to Oracle Portal Components

The two most basic building blocks of Oracle Portal are as follows:

- Portlets are the fundamental building blocks of a portal page. Portlets can be deployed to Oracle Portal through three producer (known in Oracle Portal as provider) types:
  - **Web (or Oracle PDK-Java) producers** use open standards, such as XML, SOAP, HTTP, or J2EE for deployment, definition, and communication with applications. The Oracle PDK-Java provides a framework that simplifies the task of building Web producers. For more information, see Section 58.2.3.1, "PDK-Java Producers."
  - **WSRP producers** serve as the containers for standards-based portlets. For more information, see Section 58.2.3.2, "WSRP Producers."
  - **Database producers** own one or more database portlets. Examples of database portlets include, portlets built using the Oracle PDK-PL/SQL, Portlet Builder portlets (charts, forms, reports, and so on), and Oracle Portal page portlets).

  In WebCenter Portal: Framework, you can make Web and WSRP portlets available by registering their producers with the application. For more information, see Section F.2.1, "How to Reuse JSR 286 and Oracle PDK-Java Portlets."

  You can also make database portlets available to Framework applications by using the Federated Portal Adapter. For more information, see Section F.2.8, "How to Use the Federated Portal Adapter to Reuse Database Portlets."
- Items are individual pieces of content (text, hyperlink, image, and so on) that reside on a page in an item region. Users with an appropriate privilege level can add items to a page. Item content and metadata are stored in the Oracle Portal schema of the Metadata Repository. Items are rendered on the page according to the layout, style, and attribute display defined for the item region.

Because items are stored in a content repository, you can retrieve them with Java Content Repository (JCR) data controls. A default adapter for the Oracle Portal content repository is provided with WebCenter Portal: Framework. For more information, see Section F.3, "Reusing Items."

Given the technical differences between Oracle Portal and WebCenter Portal, there is no direct upgrade available across the platform. However, key components of your portal can be exposed within WebCenter Portal.

F.2 Reusing Portlets

You can make the portlets used in your portal available to your Framework applications.

This section includes the following subsections:

- Section F.2.1, "How to Reuse JSR 286 and Oracle PDK-Java Portlets"
- Section F.2.2, "What You May Need to Know About Events"
- Section F.2.3, "What You May Need to Know About Mobile Portlets"
- Section F.2.4, "What You May Need to Know About the Portlet Chrome"
- Section F.2.5, "What You May Need to Know About Personalizations and Customizations"
- Section F.2.6, "What You May Need to Know About Oracle Portal System Resources"
- Section F.2.7, "What You May Need to Know About Partner and External Applications"
- Section F.2.8, "How to Use the Federated Portal Adapter to Reuse Database Portlets"
- Section F.2.9, "What You May Need to Know About Troubleshooting the Federated Portal Adapter"
- Section F.2.10, "What You May Need to Know About Limitations of the Federated Portal Adapter"
- Section F.2.11, "How to Reuse Oracle PDK-Java Producers from Earlier Oracle Application Server Versions"

F.2.1 How to Reuse JSR 286 and Oracle PDK-Java Portlets

To reuse JSR 286 and Oracle PDK-Java portlets in your Framework application, simply register the producers in the same way as any other producer.

To reuse JSR 286 and Oracle PDK-Java portlets:

1. In Oracle JDeveloper, access the WSRP or Oracle PDK-Java Producer Registration wizard.
2. Step through the wizard providing information about the producer, including the URL endpoint.
3. Open the page to which you to add the portlet.
4. Drag the portlet from the registered producer to the appropriate part of the page.

For more detailed information about consuming portlets in a Framework application, see Chapter 64, "Consuming Portlets."

F.2.2 What You May Need to Know About Events

Oracle PDK-Java events are not supported in WebCenter Portal: Framework. If you have Oracle PDK-Java portlets that use events and you deploy them in an WebCenter Portal: Framework environment, then the events are ignored.

F.2.3 What You May Need to Know About Mobile Portlets

Mobile portlets are not supported in WebCenter Portal: Framework. If you have Oracle PDK-Java mobile portlets, then they do not work in a WebCenter Portal: Framework environment.

F.2.4 What You May Need to Know About the Portlet Chrome

In WebCenter Portal: Framework, the portlet does not provide the chrome. The consuming application provides the chrome. This behavior conforms to the WSRP convention, but it is a change from Oracle Portal, where Oracle PDK-Java provided the chrome for portlets. Hence, the Oracle PDK-Java portlet header is filtered out in the WebCenter Portal: Framework environment.

As part of this filtering process, WebCenter Portal: Framework parses the title area of PDK-Java portlets to preserve the title and add the necessary portlet mode links. The filtering algorithm is as follows:

- If the portlet has multiple tables, then the title area is considered to be the first table. If the portlet has only one table, then the title area is considered to be the first row in that table.
- The first text in the title area is considered to be the title. If the portlet has no tables or a discernible title, then the title is taken from the `adfp:portlet` component.
- Any link that contains the fragment `_mode=` *`parameter`* is considered to be a portlet mode link. These links are moved into the portlet menu by WebCenter Portal: Framework.

If you want to bypass the rendition of the header section of your portlet chrome and reuse its original, Oracle PDK-Java header, then you can set the `displayHeader` attribute of the `adfp:portlet` tag to `false` and `retainPortletHeader` as `true` in the associated portlet bindings of the page definition XML. Setting `retainPortletHeader=true` in the page definition portlet binding retains the portlet header in the portlet response. Setting `displayHeader=false` in the `adf:portlet` tag suppresses the application's header for the portlet chrome, and hides the icons and links normally displayed in the header.

F.2.5 What You May Need to Know About Personalizations and Customizations

Portlet customizations and personalizations that you or your users applied to your portlets in Oracle Portal are not carried over to your Framework application. In a Framework application, your Oracle Portal portlets are treated as new instances. Hence, previous personalizations and customizations are not available.

F.2.6 What You May Need to Know About Oracle Portal System Resources

Portlets built with a target consumer of Oracle Portal in mind tend to make invalid assumptions about the runtime environment. For example, the portlet might assume the presence of Oracle Portal resources, such as images or JavaScript functions. In WebCenter Portal: Framework, these resources are not available. Hence, you must bundle the resources the portlet needs with the producer to ensure that the portlet runs properly in the WebCenter Portal: Framework environment.

Another common issue with portlets designed for Oracle Portal consumption is the portlet assuming the Oracle Portal URL format and the presence of Oracle Portal parameters. The page URL of an Oracle ADF application is different from that of Oracle Portal. Therefore, portlet developers cannot make assumptions about the page URL and must prepare their Oracle PDK-Java portlets for execution in the WebCenter Portal: Framework environment by using the proper portlet APIs.

F.2.7 What You May Need to Know About Partner and External Applications

Partner applications are not supported. External applications are supported. Automatic login is supported through an external application automated login servlet, which is automatically configured in your application when you register a producer.

F.2.8 How to Use the Federated Portal Adapter to Reuse Database Portlets

The Federated Portal Adapter is a component of Oracle Portal that enables Oracle Portal instances to share their database portlets through the Web portlet interface. The Federated Portal Adapter receives the SOAP messages for a Web producer, parses the SOAP and then dispatches the messages to a database producer as PL/SQL procedure calls. In effect, the Federated Portal Adapter makes a database producer behave in the same way as a Web producer. You can make Oracle Portal database portlets, including PL/SQL portlets, Portlet Builder portlets, and page portlets, available for use in your Framework applications.

> **Note:** You can also expose Oracle Portal pages in WebCenter Portal through the Federated Portal Adapter by publishing them as portlets in Oracle Portal.

Implementing the Federated Portal Adapter for your portal exposes the Oracle Portal portlets through the Oracle PDK-Java Web producer protocol making it possible to register them as PDK-Java producers. Registering the producer with a Framework application causes the portlets to appear in the Component Palette, and you can then drag and drop them onto pages in the same way as any other portlet. Without the Federated Portal Adapter, you cannot access these producers from a Framework application.

For more information about the Federated Portal Adapter, see the chapter "Using the Federated Portal Adapter" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Portal*.

To register a Federated Portal Adapter producer with WebCenter Portal:

1. Set up the Oracle Portal environment for the Federated Portal Adapter. For more information, see the section "Setting Up the Environment to Use the Federated Portal Adapter" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Portal*.
2. To register an instance of Oracle Portal through the Federated Portal Adapter, the portal must have a user named "PORTAL" in Oracle Internet Directory. To create this user:
   a. Log in to the Oracle Portal instance as the portal administrator user.
   b. In the **User** portlet, click **Create New Users.**

      By default, the **User** portlet is on the **Administer** tab of the **Portal Builder** page, on the **Portal** subtab.

c. In the **Last Name** field, enter `PORTAL`.

d. In the **User ID** field, enter `PORTAL`.

e. In the **Password** and **Confirm Password** fields, enter an appropriate password.

f. In the **Email Address** field, enter an appropriate email address.

g. Click **Submit,** then **Done.**

h. In the **Portal User Profile** portlet, enter `PORTAL` in the **Name** field, and click **Edit.**

By default, the **Portal User Profile** portlet is on the **Administer** tab of the **Portal Builder** page, on the **Portal** subtab.

i. In the **Default Group** field, enter `PORTLET_PUBLISHERS`.

j. Click **OK.**

3. If you want to display Oracle Portal pages in your Framework applications, you must first publish each of those pages as portlets. To display Oracle Portal pages, edit the page properties and select **Publish As Portlet** on the **Optional** tab. For more information, see the section "Placing One Page Onto Another" in the *Oracle Fusion Middleware User's Guide for Oracle Portal*.

4. Confirm that the users of the Framework application have appropriate privileges on the portlets and pages exposed through the Federated Portal Adapter, and that they map to the users of the Oracle Portal instance. The user name for a person on the Framework application should map to the user name of the same person in Oracle Portal. For example, JSMITH should represent the same person in both the Framework application and Oracle Portal.

5. In Oracle JDeveloper, register the producer as an Oracle PDK-Java producer using the Federated Portal Adapter URL and Service ID:

```
http://host:port/adapter/dad/schema
```

To verify the URL, enter it in a browser. If the URL is correct you should see the following message:

```
Congratulations - you got to the adapter test page
```

If you are logged in you should see a list of producers, their service IDs, and the portlets each provides. If you are logged in as an Oracle Portal administrator, you should also see additional details about page portlets and Oracle Portal HMAC registrations.

When you register the producer, ensure that you select the **Establish Producer Sessions** checkbox on the Specify Connection Details page of the wizard. This is so that the WebCenter Portal user information is correctly propagated to Oracle Portal.

For more information, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

6. You can drag and drop the producer's portlets onto a page.

7. To secure communication between the Oracle Portal instance and WebCenter Portal, provide a shared key using Hash Message Authentication Code (HMAC):

a. Go to SQL*Plus and log in as the portal schema owner.

b. Run:

```
@wwc/proadssr.sql http://www.oracle.com/adapter/portal
```

Note that, for Framework applications, the adapter URL is always the same:

```
http://www.oracle.com/adapter/portal
```

c. Enter a shared key code.

d. In Oracle JDeveloper, enter this shared key code into the **Shared Key** field on the Specify Additional Registration Details step of the Create/Edit Oracle PDK-Java Portlet Producer wizard or dialog.

Note: To remove the shared key from the Oracle Portal instance use:

```
@wwc/proadsdr.sql
```

---

8. Certain portlets built in Oracle Portal, such as page portlets and several sample database portlets, assume that the set of images shipped in the portal middle tier are available in the path `/images` in the Web server of the portal page. These images are not available by default for your Framework applications. For these images to display correctly:

   a. Download the following zip file:

   http://download.oracle.com/otndocs/tech/portal/files/portal-images.zip

   b. Extract the contents of the zip file and mount the contents under the URL `/images`.

9. You can now run the page that contains the portlet.

F.2.9 What You May Need to Know About Troubleshooting the Federated Portal Adapter

The following information may help when trying to resolve issues with the Federated Portal Adapter in WebCenter Portal.

This section includes the following subsections:

- Section F.2.9.1, "On Registration"
- Section F.2.9.2, "During Runtime"

F.2.9.1 On Registration

Error occurred trying to register producer

The shared key has not been set up in the Oracle Portal instance or it does not match that given in Oracle JDeveloper, or the service ID has not been entered.

Could not Contact Producer

Either the portal registration URL is not correct, the service ID given in Oracle JDeveloper does not match a provider in the Oracle Portal instance, the Oracle Portal instance could not be contacted (for example, it is behind a firewall or is not available), or the users in Oracle JDeveloper do not match those in Oracle Portal. The details of the message should give further information about the cause:

- 404 Not Found: The Oracle Portal instance could not be contacted or has timed out. Try increasing the time out period.

- 406 Not Acceptable: The provider with the given service ID could not be found in this Oracle Portal instance or HMAC authentication has either failed or been incorrectly configured.
- 503 Service Unavailable: the PORTAL user could not be found in Oracle Internet Directory. The Oracle Portal administrator may not have added the PORTAL user to Oracle Internet Directory.

F.2.9.2 During Runtime

Could not Contact Producer

Either the portal registration URL is not correct, the service ID given in Oracle JDeveloper does not match a provider in the Oracle Portal instance, the Oracle Portal could not be contacted (for example, if it is behind a firewall or not available), or the users in Oracle JDeveloper do not match those in Oracle Portal. The details of the message should give further information about the cause:

- 404 Not Found: The Oracle Portal instance could not be contacted or has timed out. Try increasing the time out period.
- 406 Not Acceptable: The provider with the given service ID could not be found in this Oracle Portal instance or HMAC authentication has either failed or been incorrectly configured.
- 503 Service Unavailable: the PORTAL user could not be found in Oracle Internet Directory. The Oracle Portal administrator may not have added the PORTAL user to Oracle Internet Directory.

Portlet does not render properly

- If images are missing or appear as links, see the instructions for downloading the images zip file provided in Section F.2.8, "How to Use the Federated Portal Adapter to Reuse Database Portlets."
- If navigation does not work properly, try editing the page and setting `RenderPortletInIframe` to `true`. Some portlets work better in their own inline frame (IFRAME).
- If the portlet responds with the following message, try putting the portlet in an IFRAME, by setting `RenderPortletInIframe` to `true`:

  ```
  The portlet attempted to issue a redirect in response to a render request.
  ```

- If popups, lists of values, and so on do not work in the portlet, it may be because JavaScript exceptions are raised in an attempt to avoid XSS attacks. The popup must have the same server URL as the Framework application.
- If a portlet renders a blank screen for Customize or Personalize, it may be that the user does not have permission to edit the portlet. Check the Portal log files.

F.2.10 What You May Need to Know About Limitations of the Federated Portal Adapter

The following limitations apply when using the Federated Portal Adapter with WebCenter Portal:

- Deep linking may not work. In some cases, putting the portlet in an IFRAME helps (setting `RenderPortletInIframe` to `true`).
- Links rendered by the portlet must be absolute URLs.
- If a page portlet displays a page with tabs it should be placed into an IFRAME.

- Refresh, Collapse, and Remove portlet links do not work in portlets displayed in page portlets.
- Not all objects are returned by the Federated Portal Adapter, and thus are not visible in WebCenter Portal. For a full list, see "Oracle Portal - Limitations" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

F.2.11 How to Reuse Oracle PDK-Java Producers from Earlier Oracle Application Server Versions

If you have a producer that you designed and deployed on Oracle Application Server Release 2 (10.1.2) or earlier, then you can reuse it in either one of two ways:

- You can consume a portlet from a producer running on an Oracle Application Server Release 2 (10.1.2.0.2) middle tier. In this case, the producer's code is running on Release 2 (10.1.2.0.2) while the application consuming the producer is running on Release 11.
- You can redeploy the producer application as an EAR file on Oracle WebLogic Server and consume its portlets in other Framework applications. In this case, both the producer's code and the consuming application run on Release 11.

In either of these cases, you must take some additional steps to get the producer's portlets to operate correctly. Portlets built for Oracle Portal expect certain boilerplate images to be present on the page assembly servlet and they break if those images are not available. These images were included in the middle-tier servlet by default in Oracle Application Server Release 2 (10.1.2.0.2), but for WebCenter Portal: Framework you must manually ensure that the images can be found by portlets.

This section includes the following subsections:

- Section F.2.11.1, "Consuming a Portlet from Oracle Portal"
- Section F.2.11.2, "Redeploying PDK-Java Producers from Oracle Portal"

F.2.11.1 Consuming a Portlet from Oracle Portal

In WebCenter Portal, to consume a portlet running on a page in an Oracle Portal instance, you can do either of the following:

- Obtain the Oracle Portal images zip file from the Oracle Technology Network and unzip it inside the ADFP servlet.
- Properly configure the resource servlet as follows:

1. Add an initialization parameter to the Web servlet as shown is Example F–1.

Example F–1 Web Servlet Initialization Parameters

```
<servlet>
 <servlet-name>SOAPServlet</servlet-name>
 <display-name>SOAPServlet</display-name>
 <description>Extended Portal SOAP Server</description>
 <servlet-class>oracle.webdb.provider.v2.adapter.SOAPServlet</servlet-class>
 ...
 <init-param>
   <param-name>resourceServletMapping</param-name>
   <param-value>/pdkresource</param-value>
 </init-param>
</servlet>
```

2. Add a servlet definition for the resource servlet as shown in Example F–2.

Example F–2 Servlet Definition for Resource Servlet

```
<servlet>
 <servlet-name>ResourceServlet</servlet-name>
  <display-name>ResourceServlet</display-name>
  <description>Image resource servlet</description>
  <servlet-class>oracle.webdb.provider.v2.adapter.ResourceServlet</servlet-class>
</servlet>
```

3. Add a servlet mapping for the resource servlet as shown in Example F–3.

Example F–3 Servlet Mapping for Resource Servlet

```
<servlet-mapping>
 <servlet-name>ResourceServlet</servlet-name>
 <url-pattern>/pdkresource/*</url-pattern>
</servlet-mapping>
```

4. Now you can register this PDK-Java producer in the same way as any other PDK-Java producer. For more information about registering PDK-Java producers, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

F.2.11.2 Redeploying PDK-Java Producers from Oracle Portal

If you choose to take a PDK-Java producer built for Oracle Portal and redeployed on Oracle WebLogic Server, then you must enable the resource servlet as follows:

1. Add an initialization parameter to the Web servlet as shown in Example F–4.

Example F–4 Web Servlet Initialization Parameter

```
<servlet>
 <servlet-name>SOAPServlet</servlet-name>
 <display-name>SOAPServlet</display-name>
 <description>Extended Portal SOAP Server</description>
 <servlet-class>oracle.webdb.provider.v2.adapter.SOAPServlet</servlet-class>
 ...
 <init-param>
   <param-name>resourceServletMapping</param-name>
   <param-value>/pdkresource</param-value>
 </init-param>
</servlet>
```

2. Add a servlet definition for the resource servlet as shown in Example F–5.

Example F–5 Servlet Definition for Resource Servlet

```
<servlet>
 <servlet-name>ResourceServlet</servlet-name>
  <display-name>ResourceServlet</display-name>
  <description>Image resource servlet</description>
  <servlet-class>oracle.webdb.provider.v2.adapter.ResourceServlet</servlet-class>
</servlet>
```

3. Add a servlet mapping for the resource servlet as Example F–6.

Example F–6 Servlet Mapping for Resource Servlet

```
<servlet-mapping>
 <servlet-name>ResourceServlet</servlet-name>
 <url-pattern>/pdkresource/*</url-pattern>
```

```
</servlet-mapping>
```

4. Now you can register this PDK-Java producer in the same way as any other PDK-Java producer. For more information about registering PDK-Java producers, see Section 64.2.3, "How to Register an Oracle PDK-Java Portlet Producer."

F.3 Reusing Items

Items from Oracle Portal have no equivalent in WebCenter Portal: Framework. As such, creating, maintaining, and publishing items is not supported in WebCenter Portal: Framework. To replicate the behavior of items with WebCenter Portal: Framework, you can use JCR data controls to include content from Oracle Portal. Oracle Portal is a content repository that WebCenter Portal: Framework supports out of the box in the Create a Content Repository Data Control dialog in JDeveloper. For more information about integrating content from Oracle Portal, see Chapter 27, "Managing Content Repository Connections."

G

Expression Language Expressions

This appendix catalogs the Expression Language expressions (ELs) available for use with the custom portals and applications you create using Oracle WebCenter Portal: Framework. Some of these expressions can be used in both Framework applications and the Spaces application.

This appendix includes the following sections:

- Section G.1, "ELs Related to Applications"
- Section G.2, "ELs Related to Security"
- Section G.3, "ELs Related to General Settings"
- Section G.4, "ELs Related to Application Resources"
- Section G.5, "ELs Related to Navigation"
- Section G.6, "ELs Related to WebCenter Portal Services"
- Section G.7, "ELs Related to the Documents Service"
- Section G.8, "ELs Related to the People Connections Service"
- Section G.9, "ELs Related to the Personalization Service"

> **See Also:** For additional information about EL APIs, see *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*.

G.1 ELs Related to Applications

Table G–1 lists an EL expression relevant to an application and describes the type of functionality it provides. The listed EL applies to both Framework applications and the Spaces application.

Table G–1 EL Expression Relevant to an Application

Expression	Function
`#{requestContext.skinFamily}`	Returns the name of the ADF Faces skin family being used for the current web request, depending on factors such as what has been configured at the application level, the current user's preference setting, and so on. Returns the same value as `#{adfFacesContext.skinFamily}`.

G.2 ELs Related to Security

Table G.2 lists EL expressions relevant to application security and describes the types of functionality they provide. Some of these expressions can be used in both Framework applications and the Spaces application. The **Context** column indicates whether the EL is exclusive to WebCenter Portal: Framework (WCF) or can be used with both (Both).

Table G–2 EL Expressions Relevant to Security

Expression	Function	Context
`#{security.authenticated}`	Returns `true` when the current user is authenticated in the context in which the EL is being invoked, otherwise `false`.	Both
`#{securityContext.userName}`	Returns the user name of the currently logged in user. If the current user is not logged in, this expression returns no value.	Both
`#{WCSecurityContext.currentUser['userName']}`	Returns the value `true` if the current user is the specified user, otherwise `false`.	Both
`#{WCSecurityContext.userInAppRole['role']}`	Returns the value `true` if the current user is assigned the specified OPSS application role. This expression is useful in Framework applications that are scope unaware (that is, that do not support spaces). Example: `#{WCSecurityContext.userInAppRole['AppConnectionManager']}`	WCF
`#{WCSecurityContext.userInGroup['group']}`	Returns the value *true* if the current user is assigned the specified group, for example: `#{WCSecurityContext.userInGroup['Administrators']}`	Both

G.3 ELs Related to General Settings

Table G.3 lists EL expressions relevant to general application settings and describes the types of functionality they provide. All listed ELs apply to both Framework applications and the Spaces application.

Table G–3 EL Expressions Relevant to General Settings

Expression	Function
`#{generalSettings.userTimeZone}`	Returns the time zone the current user has selected in application preferences.
`#{generalSettings.preferredDateStyle}`	Returns the date format the current user has selected in application preferences.
`#{generalSettings.preferredDatePattern}`	Returns the current user's preferred date format pattern if it has been set, else, returns `null`. Note that date + pattern is applicable only to Framework applications, where they take precedence over date + style if they are set.
`#{generalSettings.preferredTimeStyle}`	Returns the time format the current user has selected in application preferences.

Table G–3 (Cont.) EL Expressions Relevant to General Settings

Expression	Function
`#{generalSettings.preferredTimePattern}`	Returns the current user's preferred time format pattern if it has been set, else, returns `null`.
`#{generalSettings.preferredDateTimePattern}`	Returns the current user's preferred date and time format pattern if it has been set, else, returns `null`.
`#{generalSettings.preferredAccessibilityMode}`	Returns the current user's preferred accessibility mode (either `default`, `inaccessible`, or `screenReader`)
`#{generalSettings.preferredSkinName}`	The current user's preferred skin name if one is specified, otherwise the default value.
`#{generalSettings.formattedCurrentDate}`	Returns the current date in the current user's selected locale.
`#{generalSettings.formattedCurrentTime}`	Returns the current time in the current user's selected locale.
`#{generalSettings.formattedCurrentDateTime}`	Returns the current date and time in the current user's selected locale.
`#{requestContext.skinFamily}`	Returns the name of the ADF Faces skin family being used for the current web request, depending on factors such as what has been configured at the application level, the current user's preference setting, and so on.

G.4 ELs Related to Application Resources

Use the expressions in this section to query for resources. Querying for a resource through an EL expression is similar to querying for it through an API call. That is, you must set query parameters in the format `.`*`property`*`['`*`value`*`']`, where *`property`* is the name of the property, for example, `id`, `resourceScope`, and the like, and *`value`* is the search value for the attribute.

A query can result in single or multiple results. The query designer must decide what is wanted. The query designer determines whether to return one or multiple results by encountering one of the following values in the expression:

- `singleResult`—Returns a single resource. When no matching resource is found, null is returned.
- `resultList`—Returns a list of resources. When no matching resources are found, an empty list is returned.

> **Note:** Occurrences of `singleResult` or `resultList` in the expression are used internally as the query end point, and, after this, the query is executed immediately. Anything set after the end point can result in unexpected behavior.

The following example returns the first template resource found with a display name that contains `myPage`:

```
#{srmContext.resourceType['siteTemplate'].displayName['myPage'].singleResult}
```

The following example returns a list of template resources residing in the directory `resourceDir`, with a description that contains `sampleDesc`:

```
#{srmContext.resourceType['siteTemplate'].description['sampleDesc'].contentDirecto
```

```
ry['resourceDir'].resultList}
```

A property of this class includes any attribute of this class. Example properties include: `Id`, `DisplayName`, `iconURI`, `contentDirectory`, and the like.

The property value can be an explicit value or an EL expression that returns that type of value. For example, the following two queries return the same result:

```
#{srmContext.id['resourceId'].singleResult}

#{srmContext.id['spacesContext.currentSpace.uiMetadata.siteTemplateId'].singleResu
lt}
```

You can use any property of this class in an EL-based query in the format *`property`*`['`*`value`*`']` and in any order. For example, the following two queries return the same result:

```
#{srmContext.resourceScope['scopeName'].id['resourceId'].singleResult}

#{srmContext.id['resourceId'].resourceScope['scopeName'].singleResult}
```

Table G–4 lists EL expressions relevant to portal resources and describes the types of functionality each provides. You can use all resource EL expressions in both Spaces and Framework applications built with Oracle JDeveloper.

Many of the expressions in Table G–4 are building blocks for narrowing down the resource or resources you want returned. That is, they are not necessarily meant to be used by themselves, but rather in an assembly of ELs to form the desired query.

For example, the following expression uses three resource-related ELs to form a single query that returns a particular space template. It retrieves the template `storesMasterTemplate` from the parent space `stores`:

```
#{srmContext.resourceScope['stores'].resourceType['siteTemplate'].displayName['sto
resMasterTemplate'].singleResult}
```

Table G–4 ELs Relevant to Portal Resources

EL	Function
`#{srmContext.id['`*`resourceGUID`*`']}`	Returns the resource with the specified ID
`#{srmContext.displayName['`*`resourceDisplayName`*`' ]}`	Returns any resources with the specified display name
`#{srmContext.displayNameKey['`*`displayNameKey`*`']}`	Returns any resources with the specified display name key
`#{srmContext.description['`*`resourceDescription`*`' ].singleResult}`	Returns one result of a resource that contains the specified value in its description To get multiple results, use `resultList` in lieu of `singleResult`.
`#{srmContext.descriptionKey[`*`descriptionKey`*`].si ngleResult}`	Returns one result of a resource with the specified description key The description key is the key in the `xsrt` file for the resource description. To get multiple results, use `resultList` in lieu of `singleResult`.
`#{srmContext.createdDate['`*`resourceCreationDate`*` ']}`	Returns any resource with the specified creation date

Table G–4 (Cont.) ELs Relevant to Portal Resources

EL	Function
`#{srmContext.modifiedBy['resourceLastModifiedBy']}`	Returns any resource that was last modified by the user with the specified user name
`#{srmContext.modifiedDate['resourceLastModifiedDate']}`	Returns any resource that was last modified by the specified date
`#{srmContext.resourceScope['resourceScope']}`	Returns any resource that falls within the specified scope
`#{srmContext.category['categoryName']}`	Returns any resource that falls within the specified category For example: `#{srmContext.category['siteTemplates']}`
`#{srmContext.contentDirectory['contentDirectory']}`	Returns any resource that is stored within the specified directory
`#{srmContext.metadataFile['metadataFileLocation']}`	Returns resource metadata from the specified metadata file For example, the following expression returns resource metadata from the following file `/home/metadata/data.xml`: `#{srmContext.metadataFile['/home/metadat/data.xml']}`
`#{srmContext.jspx['jspxFileLocation']}`	Returns any `jspx` file in the specified location For example, the following expression returns the `page.jspx` file: `#{srmContext.jspx['/home/web/page.jspx']}`
`#{srmContext.pageDef['pageDefinition']}`	Returns any `jspx` file with the specified page definition
`#{srmContext.iconURI['iconURI']}`	Returns the icon at the specified icon URI
`#{srmContext.excludeFrom['excludeFromScopes']}`	In a larger expression, returns all specified resources except those available in the excluded scopes
`#{srmContext.usesCustomSecurity['usesCustomSecurity']}`	In a larger expression, returns any resource the either does or does not use custom security Set *usesCustomSecurity* to `TRUE` or `FALSE`. For example: `#{srmContext.usesCustomSecurity['TRUE']}`
`#{srmContext.seeded['seeded']}`	In a larger expression, returns any resource that is or is not seeded, depending on the provided value Set *seeded* to `TRUE` or `FALSE`. For example: `#{srmContext.seeded['TRUE']}`

Table G–4 (Cont.) ELs Relevant to Portal Resources

EL	Function
`#{srmContext.visibleType['visibleType']}`	In a larger expression, returns any resource that is of the specified type of visibility Set *visibleType* to `TRUE`, `FALSE`, or `NEVER` to indicate that the resource is never shown. For example, the following expression returns resources that are visible, that is, resources that are set to **Show**: `#{srmContext.visibleType['TRUE']}`
`#{srmContext.visible['visible']}`	In a larger expression, returns one or multiple resources with visibility set to either `TRUE` or `FALSE`
`#{srmContext.createdBy['resourceCreator']}`	In a larger expression, returns any resource created by the specified user
`#{srmContext.resourceType['resourceType']}`	In a larger expression, returns one or multiple resources of the specified type For example, the following expression searches for a resource of the type `SITE_TEMPLATE`: `#{srmContext.resourceType['SITE_TEMPLATE']}` Note that all resource types are listed in *Oracle Fusion Middleware Java API Reference for Oracle WebCenter Portal*, in the `GenericSiteResourceTypes` class.
`#{srmContext.version['version']}`	In a larger expression, returns one or multiple resources available in the application of the specified version
`#{srmContext.resourceScope['scopeName']}`	In a larger expression, returns one or multiple resources available in the specified scope For example, the following expression searches for resources in the scope (in this instance, the space) `MySpace`: `#{srmContext.resourceScope['MySpace']}` To search in the default scope, that is, the application scope, use `defaultScope`.
`#{srmContext.searchType['searchType']}`	In a larger expression, returns one or multiple resources that contain or equal the values set by other included expressions Set *searchType* to `CONTAINS` or `EQUALS`.

G.5 ELs Related to Navigation

Table G.5 lists EL expressions relevant to application navigation and describes the types of functionality they provide. All listed navigation ELs apply to both Framework applications and the Spaces application.

Table G–5 EL Expressions Relevant to Navigation

Expression	Function
Navigation Context Expressions	
`#{navigationContext.defaultNavigationModel}`	Returns default navigation model. It gets the value from the preference bean. The preference bean gets the value based on the preference setting in `adf-config.xml`, `static value`. The following example returns the specified default tree model: `#{navigationContext.defaultNavigationModel.defaultTreeModel}` You can also use `defaultMenuModel` and `defaultListModel` (see next listing).
`#{navigationContext.currentNavigationModel}`	Returns the navigation model used to navigate to the current view (that is, the current page) There may be a number of navigation models on the page. Only the one used to navigate to the page is the current navigation model. The current navigation model is set only when the `processAction` is called. The following example returns the default tree model for the current navigation model: `#{navigationContext.currentNavigationModel.defaultTreeModel}` You can also use `defaultMenuModel` and `defaultListModel`. **See Also:** `#{navigationContext.processAction}`.

`#{navigationContext.navigationModel['model_path']}`	Returns the navigation model specified by path.
	Example:
	`#{navigationContext.navigationModel['/oracle/webcenter/portalapp/navigations/default-navigation-model']}`
	User can simply enter the path in the square bracket without having to remember the parameter name. For backward compatibility, comma-separated parameters are also supported. Available parameters (with examples) are:
	■ `modelPath=/oracle/webcenter/portalapp/navigations/default-navigation-model`
	■ `modelScope=/oracle/webcenter/portalapp/navigation`
	■ `modelId=default-navigation-model`
	Note: `modelScope` and `modelId` are used together, for example:
	`#{navigationContext.navigationModel['modelScope=/oracle/webcenter/portalapp/navigations,modelId=default-navigation-model']}`

Table G–5 (Cont.) EL Expressions Relevant to Navigation

Expression	Function

`#{navigationContext.processAction}`	Returns the default navigation method for binding to UI component's `actionListener` attribute. This assumes that the target resource to navigate to is passed in through the action UI component's `node` or `path` attribute.
	
	The `node` attribute is typically used for the iterative case; while the `path` attribute is typically used to create a static link. If both are specified, the `node` attribute is used. Note that when using the `path` attribute, you must also specify the `model` attribute to pass in the navigation model object. If the `model` attribute is not specified, the current navigation model is used.
	
	Example using the `node` attribute:
	
	`<af:forEach var="node" varStatus="vs"`
	`  items="#{navigationContext.defaultNavigationModel.rootNode.children}">`
	`  <af:subform id="pt_sfm1">`
	`  <af:commandLink id="pt_cl1"`
	`  text="#{node.title}"`
	`  inlineStyle="font-size:small;color:White;"`
	`  actionListener="#{navigationContext.processAction}"`
	`  action="pprnav">`
	`<f:attribute name="node" value="#{node}"/>`
	`  <af:showPopupBehavior popupId="menuPopup"`
	`  align="afterStart"`
	`  triggerType="mouseOver"/>`
	`  </af:commandLink>`
	
	Example using the `path` attribute:
	
	`<af:commandLink id="pt_cl1" text="Prescriptions"`
	`  inlineStyle="font-size:small; color:White;"`
	`  actionListener="#{navigationContext.processAction}"`
	`  action="pprnav">`
	`<f:attribute name="path" value="/prescriptions"/>`
	`<f:attribute name="model"`
	`  value="#{navigationContext.defaultNavigationModel}"`
	`/>`
	`</af:commandLink>`

Navigation Model Expressions	

`#{navigationContext.defaultNavigationModel.defaultTreeModel}`
 	
`#{navigationContext.defaultNavigationModel.defaultMenuModel}`
 	
`#{navigationContext.defaultNavigationModel.defaultListModel}`
 	
`#{navigationContext.defaultNavigationModel.defaultSiteMap}`
	Returns a tree/menu/list model based on the default settings
	
	The default values for the settings are specified in the next row. For example, the default value for `depth` is `0`.
	
	For `defaultSiteMap`, it returns the XML for the site map of the navigation based on the default settings.
	
	This expression returns a model of type:
	
	■ `ChildPropertyTreeModel`—used in `<af:tree>` component
	
	■ `ChildPropertyMenuModel`—used in `<af:breadcrumbs>` or <af:menu> component
	
	■ `List`*`NavigationResource`*—used in `<af:foreach>`

Table G–5 (Cont.) EL Expressions Relevant to Navigation

Expression	Function

`#{navigationContext.defaultNavigationModel.treeModel['parameters']}`
`#{navigationContext.defaultNavigationModel.menuModel['parameters']}`
`#{navigationContext.defaultNavigationModel.listModel['parameters']}`
`#{navigationContext.defaultNavigationModel.siteMap['parameters']}`
	Returns tree/menu/list model based on the specified comma-separated parameters. Available parameters (with default values as examples) are:
	■ `startNode=/`—specify the starting node of the model (do not need "/" prefix unless requesting the root node, for example, `home`).
	■ `includeStartNode=true`—specify `true` if you want to include the starting node (for example, the root node above) or `false` to start from its children.
	■ `depth=0`—defines the initial depth of fetching. "0" means fetch the entire tree, which may take a long time. In which case, use "1" to fetch on demand (when users click the **Expand** icon).
	■ `prefetchOnly=false`—by default (`true`), it returns nodes up to the depth level requested initially; deeper level nodes are returned on demand. Use `false` if you want only the initial sets of nodes. In which case, it does not return deeper nodes later on when requested, even when there are deeper nodes.
	For `siteMap`, the available parameters are `startNode` and `includeStartNode`.
	Example:
	`#{navigationContext.defaultNavigationModel.treeModel['startNode=home,includeStartNode=false,depth=2']}`

`#{navigationContext.defaultNavigationModel.rootNode}`	Returns a root node (of type `NavigationResource`) of the navigation model. Example: `<af:commandLink id="pt_cl1" text="Prescriptions" inlineStyle="font-size:small; color:White;" actionListener="#{navigationContext.processAction}" action="pprnav"> <f:attribute name="node" value="#{navigationContext. defaultNavigationModel. rootNode.children[2]"/> </af:commandLink>`
`#{navigationContext.defaultNavigationModel.node['path']}`	Returns a node (of type `NavigationResource`) based on the path specified (you do not need "/" prefix unless you are requesting the root node, for example, `'/'`) Example: `#{navigationContext.defaultNavigationModel.node['home/page1']}`
`#{navigationContext.defaultNavigationModel.currentSelection}`	Returns currently selected navigation resource.

Table G–5 (Cont.) EL Expressions Relevant to Navigation

Expression	Function

`#{navigationContext.defaultNavigationModel.newCurrentSelection['path']}`	Sets the current selection to a node specified by the given path (you do not need "/" prefix unless you are requesting the root node, for example, `'/'`). If successful, returns the newly selected node of type `NavigationResource`.
	The user must have the ability to explicitly set the current selection without having to actually navigate to a node.
	This can be used where the user enters a page directly, and no selection is set. It provides a mechanism for the user to control what is the default when there is no current selection.
	Example:
	`<c:set value="${navigationContext.defaultNavigationModel.newCurrentSelection['home/page1']}" var="currSel" scope="session"/>`
	`<c:if test="${currSel!=null}">`
	`<af:forEach items="#{currSel.children}" var="cnode" varStatus="cnodestatus">`
	`...`
	`</c:if>`

`#{navigationContext.defaultNavigationModel.attributes.Description}` `#{navigationContext.defaultNavigationModel.attributes['Description']}`	Returns the value of the specified attribute of the navigation model.
`#{navigationContext.defaultNavigationModel.properties['propertyName']}`	Returns the value of the specified property of the navigation model, where propertyName is either `rootNode` or `currentSelection`.
Navigation Resource Expressions	
`#{node.attributes.Description}` `#{node.attributes['Description']}`	Returns the value of the specified attribute of the navigation resource.
`#{node.parameters.StockSymbol}` `#{node.parameters['StockSymbol']}`	Returns the value of the specified parameter of the navigation resource.
`#{node.parametersRaw.StockSymbol}` `#{node.parametersRaw['StockSymbol']}`	Returns the raw value of the specified parameter of the navigation resource (before it is evaluated).
`#{node.title}`	Returns the title of the navigation resource.
`#{node.path}`	Returns the path to the navigation resource.
`#{node.externalURL}`	Returns the URL for the navigation resource of type `External Link`.
`#{node.prettyUrl}`	Returns the identifying path for this navigation resource.
`#{node.prettyUrlPath}` `#{node.prettyUrlPath[N]}`	Returns a collection of identifying paths by depth to enable its use as a starting path to drive another navigation view. For example, assuming the `currentSelection` is `home/company/products/applications`, the following EL returns `home/company/products`: `#{node.prettyUrlPath[3]}`

Table G–5 (Cont.) EL Expressions Relevant to Navigation

Expression	Function
#{*node*.goLinkPretty Url}	Returns the identifying path for this navigation resource suitable for `goLink` destination `<af:goLink id="pt_gl2" text="#{node.title}" destination="#{node.goLinkPrettyUrl}" targetFrame="#{node.attributes['Target']}" inlineStyle="font-size:small;#{node.selected ? 'font-weight:bold;' : ''}"/>`
#{*node*.separator ? ... : ...}	Returns whether this resource is a separator item.
#{*node*.navigable ? ... : ...}	Returns whether it is possible to navigate to this resource.
#{*node*.selected ? ... : ...}	Returns whether this resource is currently selected.
#{*node*.currentlySelected ? ... : ...}	Returns whether the resource is the currently selected resource and the model is the currently selected model.
#{*node*.onSelectedPath ? ... : ...}	Returns whether this node lies on the selected path. This is useful for highlighting the selected tab, for example. For example, assuming the currently selected node is `home/company/products/applications`, the following EL highlights the `home` tab. `<c:set value="${navigationContext.defaultNavigationModel.node['home']}" var="home" scope="session"/> <af:commandImageLink id="cil1" icon="#{(home.onSelectedPath) ? '/images/caremark/nav/HomeSelected.png' : '/images/caremark/nav/HomeIcon.png'}" actionListener="#{navigationContext.processAction}" action="pprnav"> <f:attribute name="node" value="#{home}"/> </af:commandImageLink>`
#{*node*.leaf ? ... : ...}	Returns whether this resource is a leaf node.
#{*node*.parent}	Returns the parent node (of type `NavigationResource`) of this node. For the root node, `null` is returned.
#{*node*.ancestors}	Returns the hierarchy list of ancestors of this node (of type NavigationResource) starting with the root node. For example, assuming the current node is `home/company/products/applications`, the following ELs return the following values: ■ `#{node.ancestors[1]}` returns the node for `home` ■ `#{node.ancestors[3]}` returns the node for home/company/products
#{*node*.depth}	Returns the depth of this node from the root node. Root node has a depth of zero.
#{*node*.siblings}	Returns the list of siblings (of type `NavigationResource`) of this node (inclusive).

Table G–5 (Cont.) EL Expressions Relevant to Navigation

Expression	Function

#{*node*.nextSibling}	Returns the next sibling in the list (of type NavigationResource) of this node.
	For example, the following code ensures that you do not output two separators in a row:
	`<c:if test="${(node.separator) && (!node.nextSibling.separator}">`
	`  <af:separator id="s166"/>`
	`</c:if>`

#{*node*.previousSibling}	Returns the previous sibling in the list (of type `NavigationResource`) of this node.
#{*node*.index}	Returns the zero-relative index of this node relative to its siblings.
#{*node*.children}	Returns a collection of child resources.
#{*node*.children[*N*].title}	
#{*node*.children[*N*].children[*N*].title}	
#{*node*.childCount}	Returns the number of children of this node.
#{*node*.childByIndex['index']}	Returns the child node (of type `NavigationResource`) specified by the zero-relative index. If not found, this returns `null`.
#{*node*.childByPath['admin']}	Returns the child node (of type `NavigationResource`) specified by the path relative to this node. The path can be deeper than one level and does not need the `'/'` prefix. If not found, this returns `null`.
#{*node*.properties['propertyName']}	Returns the value of the specified property of the navigation resource, where propertyName is one of the following: `separator`, `title`, `prettyUrl`, `prettyUrlPath`, `depth`, `leaf`, `childCount`, `navigable`, `selected`, or `currentlySelected`.

G.6 ELs Related to WebCenter Portal Services

Table G.6 lists EL expressions relevant to WebCenter Portal services and describes the types of functionality they provide. All listed ELs apply to both Framework applications and the Spaces application.

Table G–6 EL Expressions Relevant to WebCenter Portal Services

Expression	Function
`#{webcenterService['serviceId']}`	An `oracle.webcenter.framework.service.Service` object representing the WebCenter Portal service with the service ID `serviceId`. For example, the following EL returns *Documents Service*: `#{webcenterService['oracle.webcenter.doclib']}` For service IDs, see Table G–7.
`#{webcenterService['serviceId'].configured}`	Returns a Boolean value that indicates whether the WebCenter Portal service with the service ID *serviceId* is configured for use in the current Framework application. For example, the following EL returns `true` if Discussions service can be used in the application, `false` otherwise: `#{webcenterService['oracle.webcenter.collab.forum'].configured}` For service IDs, see Table G–7.
`#{sessionContext['oracle.webcenter.collab.forum'].groupInfo['spaceName'].forumId}`	Returns the forum ID of the specified space discussion forum. Enter the space name in lieu of *spaceName*.
`#{sessionContext['oracle.webcenter.collab.forum'].groupInfo['spaceName'].categoryId}`	Returns the category ID of the specified space discussion forums. Enter the space name in lieu of *spaceName*.

Table G–7 lists service IDs associated with WebCenter Portal services.

Table G–7 Service IDs

Service	ID
Announcements	oracle.webcenter.collab.announcement
Discussions	oracle.webcenter.collab.forum
Documents and Wikis	oracle.webcenter.doclib
Events	oracle.webcenter.collab.calendar.community
Instant Messaging and Presence (IMP)	oracle.webcenter.collab.rtc
Links	oracle.webcenter.relationship
Lists	oracle.webcenter.list
Mail	oracle.webcenter.collab.mail
Notifications	oracle.webcenter.notification
Page	oracle.webcenter.page
People Connections: Activity Stream	oracle.webcenter.activitystreaming
People Connections: Connections	oracle.webcenter.peopleconnections.connections
People Connections: Feedback	oracle.webcenter.peopleconnections.kudos
People Connections: Message Board	oracle.webcenter.peopleconnections.wall
People Connections: Profile	oracle.webcenter.peopleconnections.profile
Polls	oracle.webcenter.collab.survey

Table G–7 (Cont.) Service IDs

Service	ID
Recent Activities	oracle.webcenter.recentactivity
Activity Graph	oracle.webcenter.activitygraph
RSS	oracle.webcenter.rss
Search	oracle.webcenter.search
Tags	oracle.webcenter.tagging
Blogs	oracle.webcenter.blog
Worklist	oracle.webcenter.worklist

G.7 ELs Related to the Documents Service

Table G–8 lists EL expressions relevant to the Documents service and describes the types of functionality they provide. All listed ELs apply to both Framework applications and the Spaces application.

Table G–8 EL Expressions Relevant to the Documents Service

Expression	Function
`#{documentsService.defaultConnectionName}`	Gets the default content repository connection name. Returns `null` if no connection name is defined.
`#{documentsService.configured}`	Checks whether the Documents service is configured. Returns `true` if the Documents service is configured, otherwise `false`.

G.8 ELs Related to the People Connections Service

Table G–9 lists EL expressions relevant to the Profile feature of the People Connections service and describes the types of functionality they provide. All listed ELs apply to both Framework applications and the Spaces application..

Note: The entry `securityContext.userName`, included in every Profile expression, returns the name of the current user. Note also that information is returned only if it is present in the user's profile. If the information is not included in the profile, a null value is returned.

Table G–9 EL Expressions Relevant to the People Connections Service (Profile)

Expression	Function
`#{WebCenterProfile[userName].ManagerDisplayName}` `#{WebCenterProfile[securityContext.userName].ManagerDisplayName}`	The display name of the specified user's manager if the current user is allowed to know it.
`#{WebCenterProfile[userName].employeeNumber}` `#{WebCenterProfile[securityContext.userName].employeeNumber}`	The specified user's employee number if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessPOBox}` `#{WebCenterProfile[securityContext.userName].businessPOBox}`	The post office box number associated with the specified user if the current user is allowed to know it.
`#{WebCenterProfile[userName].timeZone}` `#{WebCenterProfile[securityContext.userName].timeZone}`	The time zone in which the specified user's home office is located if the current user is allowed to know it.
`#{WebCenterProfile[userName].description}` `#{WebCenterProfile[securityContext.userName].description}`	A description of the specified user (from Profile "About Me") if the current user is allowed to know it.
`#{WebCenterProfile[userName].department}` `#{WebCenterProfile[securityContext.userName].department}`	The department to which the specified user belongs if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessPager}` `#{WebCenterProfile[securityContext.userName].businessPager}`	The specified user's business pager number if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessCity}` `#{WebCenterProfile[securityContext.userName].businessCity}`	The city in which the specified user is located if the current user is allowed to know it.
`#{WebCenterProfile[userName].maidenName}` `#{WebCenterProfile[securityContext.userName].maidenName}`	The specified user's surname or last name before marriage if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessFax}` `#{WebCenterProfile[securityContext.userName].businessFax}`	The specified user's business fax number if the current user is allowed to know it.
`#{WebCenterProfile[userName].dateofHire}` `#{WebCenterProfile[securityContext.userName].dateofHire}`	The specified user's date of hire if the current user is allowed to know it.
`#{WebCenterProfile[userName].nameSuffix}` `#{WebCenterProfile[securityContext.userName].nameSuffix}`	Additional information appended to the user's name, such as `Esq.`, `Jr.`, and so on if the current user is allowed to know it.
`#{WebCenterProfile[userName].middleName}` `#{WebCenterProfile[securityContext.userName].middleName}`	The specified user's middle name if the current user is allowed to know it.

Table G–9 (Cont.) EL Expressions Relevant to the People Connections Service (Profile)

Expression	Function
`#{WebCenterProfile[userName].homePhone}` `#{WebCenterProfile[securityContext.userName].homePhone}`	The specified user's home phone number if the current user is allowed to know it.
`#{WebCenterProfile[userName].employeeType}` `#{WebCenterProfile[securityContext.userName].employeeType}`	The specified user's employee type classification, for example, `IC4` if the current user is allowed to know it.
`#{WebCenterProfile[userName].lastName}` `#{WebCenterProfile[securityContext.userName].lastName}`	The specified user's surname or last name if the current user is allowed to know it.
`#{WebCenterProfile[userName].dateofBirth}` `#{WebCenterProfile[securityContext.userName].dateofBirth}`	The specified user's birthday if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessState}` `#{WebCenterProfile[securityContext.userName].businessState}`	The state in which the specified user's home office is located if the current user is allowed to know it.
`#{WebCenterProfile[userName].homeAddress}` `#{WebCenterProfile[securityContext.userName].homeAddress}`	The specified user's home street address if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessStreet}` `#{WebCenterProfile[securityContext.userName].businessStreet}`	The street on which the specified user's home office is located if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessPostalCode}` `#{WebCenterProfile[securityContext.userName].businessPostalCode}`	The specified user's postal or ZIP code if the current user is allowed to know it.
`#{WebCenterProfile[userName].initials}` `#{WebCenterProfile[securityContext.userName].initials}`	The specified user's initials if the current user is allowed to know it.
`#{WebCenterProfile[userName].firstName}` `#{WebCenterProfile[securityContext.userName].firstName}`	The specified user's first name if the current user is allowed to know it.
`#{WebCenterProfile[userName].organizationalUnit}` `#{WebCenterProfile[securityContext.userName].organizationalUnit}`	The organizational unit to which the specified user belongs, for example, `D10`, if the current user is allowed to know it.
`#{WebCenterProfile[userName].wirelessAcctNumber}` `#{WebCenterProfile[securityContext.userName].wirelessAcctNumber}`	The specified user's wireless account number if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessPhone}` `#{WebCenterProfile[securityContext.userName].businessPhone}`	The specified user's business telephone number if the current user is allowed to know it.

Table G–9 (Cont.) EL Expressions Relevant to the People Connections Service (Profile)

Expression	Function
`#{WebCenterProfile[userName].businessCountry}` `#{WebCenterProfile[securityContext.userName].businessCountry}`	The country in to which the specified user is assigned if the current user is allowed to know it.
`#{WebCenterProfile[userName].preferredLanguage}` `#{WebCenterProfile[securityContext.userName].preferredLanguage}`	The specified user's preferred language if the current user is allowed to know it.
`#{WebCenterProfile[userName].displayName}` `#{WebCenterProfile[securityContext.userName].displayName}`	The specified user's display name if the current user is allowed to know it.
`#{WebCenterProfile[userName].userName}` `#{WebCenterProfile[securityContext.userName].userName}`	The specified user's actual name if the current user is allowed to know it.
`#{WebCenterProfile[userName].title}` `#{WebCenterProfile[securityContext.userName].title}`	The specified user's job title if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessEmail}` `#{WebCenterProfile[securityContext.userName].businessEmail}`	The specified user's business email address if the current user is allowed to know it.
`#{WebCenterProfile[userName].organization}` `#{WebCenterProfile[securityContext.userName].organization}`	The organzation to which the specified user belongs, for example, `Sales`, if the current user is allowed to know it.
`#{WebCenterProfile[userName].businessMobile}` `#{WebCenterProfile[securityContext.userName].businessMobile}`	The specified user's cell or mobile phone number if the current user is allowed to know it.
`#{WebCenterProfile[userName].expertise}` `#{WebCenterProfile[securityContext.userName].expertise}`	The specified user's business expertise if the current user is allowed to know it.

G.9 ELs Related to the Personalization Service

Table G–10 lists EL expressions relevant to the Personalization+ service and describes the types of functionality they provide. All listed ELs apply to both Framework applications and the Spaces application.

Table G–10 EL Expressions Relevant to the Personalization Service

Expression	Function

`#{p13nContext.reset}`	Clears the state of the context object in which it is invoked.
	Other examples:
	`#{p13nContext.conductor['ConductorConnection'].reset}`
	`#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].reset}`
	`#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].reset}`

`#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].update}`	Updates the property set with new values if bound to a form. The `update` method applies only to the context `PropertySetContext`, and can be called only as part of a form-based action.

`#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].results}`	Retrieves the results of the context object. In the case of `PropertySetContext`, it retrieves the property set by name and property set definition, if they exist.
	In the case of `ScenarioExecutionContext` and `ParameterizedScenarioExecutionContext`, the scenario results are retrieved.
	Other examples:
	`#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].results}`
	`#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].withInput['input1=val1;input2=value2;input3=value3'].results}`

`#{p13nContext.conductor['PropertiesConnection'].namespaces['MyNamespace'].scenario['MyScenario'].results}`	Executes a scenario by name.

Table G–10 (Cont.) EL Expressions Relevant to the Personalization Service

Expression	Function
`#{p13nContext.conductor['ConductorConnection'].namespaces['MyNamespace'].scenario['MyScenario'].withInput['input1=val1;input2=value2;input3=value3'].results}`	Executes a scenario by name with input parameters. Input parameters must be in the following format: *<paramOne>=<valueOne>;<paramTwo>=<valueTwo>;<paramThree>=<valueThree>* Where each parameter name and value is separated by semicolon. The `withInput` method applies only to the context `ScenarioExecutionContext`.
`#{p13nContext.conductor['ConductorConnection'].namespaces[myManagedBean.namespace].scenario[myManagedBean.scenario].isError}`	Determines if an error has occurred in the current context object. The `isError` method applies only to `PropertySetContext` and `~ScenarioExecutionContext`. Other examples: `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].isError}`
`#{p13nContext.conductor['ConductorConnection'].namespaces['myManagedBean.namespace'].scenario['myManagedBean.scenario'].errorMessage}`	Returns the error message if an error has occurred in the current context object. The errorMessage method applies only to `PropertySetContext` and `~ScenarioExecutionContext`. Other examples: `#{p13nContext.properties['PropertiesConnection'].namespaces['MyNamespace'].setDefinitions['MyPropertySetDefinition'].set['MyPropertySet'].errorMessage}`

H

Manually Migrating a Framework Application

This appendix explains how to manually migrate a PS2 WebCenter custom application to a PS5 WebCenter Portal: Framework application.

This appendix includes the following sections:

- Section H.1, "When is Migration Required?"
- Section H.2, "Anatomy of a Framework Application"
- Section H.3, "Overview of the Migration Procedure"
- Section H.4, "Preparing for the Migration"
- Section H.5, "Editing Configuration Files"
- Section H.6, "Modifying the Project Configurations"
- Section H.7, "Creating Portal-Specific Files"
- Section H.8, "Applying Security Configurations"

H.1 When is Migration Required?

Oracle strongly recommends that you perform the procedures in this appendix only after careful consideration and then only if absolutely necessary. The recommended best practice for migration is to create a new (PS5) Framework application using the WebCenter Portal – Framework Application template and then manually migrate the content, configurations, and logic from your old PS2 application to the new Framework application. If it is not possible to follow this recommended practice, then follow the procedures described in this appendix to migrate your application.

It is important to understand that this appendix is about manually adding WebCenter Portal: Framework file structures, components, and configurations to an existing WebCenter PS2 application that has been migrated to PS5 using the automatic migration feature in JDeveloper. Oracle WebCenter Portal: Framework features include page navigations, page hierarchies, and other features, as discussed in Chapter 9, "Understanding Framework Applications."

As explained in detail in this appendix, you first need to perform an automatic migration to PS5 (by opening your PS2 application in a PS5 version of JDeveloper). Then, you can follow the directions in this appendix to manually add portal features.

The procedure described in this appendix is optional. Even if you perform the automatic migration by opening your PS2 application in PS5 JDeveloper, your existing, you only need to perform the migration steps described in this appendix if you want to use the Oracle WebCenter Portal: Framework features.

For detailed information on the WebCenter Portal: Framework Application template and creating a WebCenter Portal: Framework application, see Chapter 5, "Creating Framework and Portlet Producer Applications."

H.2 Anatomy of a Framework Application

Besides explaining how to manually migrate an application, this appendix outlines in technical detail the configuration elements that distinguish a PS2 custom application from a PS5 Framework application. One way to read this appendix is as a detailed roadmap or "anatomy" of the structure of a Framework application versus a PS2 custom application.

For more information on the portal-specific features, see Chapter 9, "Understanding Framework Applications." For more information on the structure of a portal, see Section 9.12, "How are WebCenter Portal Files Organized?."

H.3 Overview of the Migration Procedure

The manual migration of a PS2 WebCenter custom application to a PS5 Framework application can be broken down into these basic steps:

1. Preparing for the migration.

   Preparation includes properly installing the WebCenter and JDeveloper PS5 software and opening the PS2 application in the PS5 version of JDeveloper (with PS5 WebCenter installed). See Section H.4, "Preparing for the Migration."

2. Editing configuration files.

   You must manually open and edit several configuration files to add PS5 features. See Section H.5, "Editing Configuration Files."

3. Adding project configurations.

   Project configurations include adding appropriate JAR files and technology scopes to your application. See Section H.6, "Modifying the Project Configurations."

4. Creating portal-specific files.

   You must manually create the directory structure of a Framework application and populate it with portal-specific files, like navigation, registry, page hierarchy, and catalog files. See Section H.7, "Creating Portal-Specific Files."

5. Applying security configurations.

   Finally, you need to configure ADF security and populate the `jazn-data.xml` file with appropriate policy information. See Section H.8, "Applying Security Configurations."

H.4 Preparing for the Migration

Before performing the rest of the migration steps described in this appendix:

1. Install the PS5 version of Oracle JDeveloper and WebCenter Portal's extension for Oracle JDeveloper. For details, see Section 3.2, "Installing WebCenter Portal's Extension for Oracle JDeveloper."

2. Back up your PS2 application.

3. In JDev (with the PS5 version of the WebCenter Portal extension installed), open the application:

1. From the File menu select **Open**.
2. In the Open dialog, navigate to the application `.jws` file and click **OK**.
3. If prompted to migrate the application, click **Yes**. This step migrates the application from PS2 technology to PS5 technology without changing the structure of the application. Your PS2 application will continue to function as it did in PS2.

4. Follow the migration steps described in the rest of this chapter.

H.5 Editing Configuration Files

This section lists the set of configuration files that you must manually update.

- Section H.5.1, "Updating DataBindings.cpx"
- Section H.5.2, "Updating weblogic.xml"
- Section H.5.3, "Updating faces-config.xml"
- Section H.5.4, "Updating trinidad-config.xml"
- Section H.5.5, "Updating web.xml"
- Section H.5.6, "Updating adf-config.xml"

H.5.1 Updating DataBindings.cpx

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Application Sources > DEFAULT_PACKAGE > DataBindings.cpx

Location on the filesystem:

- *PROJECT_ROOT*/adfmsrc/*DEFAULT_PACKAGE*/DataBindings.cpx

Detailed Instructions:

The following two attributes must exist in the root `<Application>` element of `DataBindings.cpx`. These attributes enable dynamic page mapping, so that any pages created at runtime (for example, through the Page Hierarchy editor) do not need to have page definition mappings in `DataBindings.cpx`.

```
<Application PageMapClass="oracle.jbo.uicli.mom.DynamicPageMapImpl"
     BasePageDefPackageName="pageDefs" ... >
```

H.5.2 Updating weblogic.xml

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > WEB-INF > weblogic.xml

Location on the filesystem:

- *PROJECT_ROOT*/public_html/WEB-INF/weblogic.xml

Detailed Instructions:

Add the following entry as a child of the root `<weblogic-web-app>` element. This element adds a reference to the REST shared library, which is necessary for the application to be able to use REST features.

```
<library-ref>
```

```
    <library-name>jaxrs-framework-web-lib</library-name>
</library-ref>
```

H.5.3 Updating faces-config.xml

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > WEB-INF > faces-config.xml

Location on the filesystem:

- *PROJECT_ROOT*/public_html/WEB-INF/faces-config.xml

Detailed Instructions:

Add the following entries as children of the root `<faces-config>` element or merge them in with any existing elements, as appropriate.

- The `CustomViewHandler` interprets the URL syntax used by the navigation model. The handler takes a URL that points to a node in a navigation and finds the corresponding JSPX page to render.
- The `SkinPhaseListener` enables any skins that are created and registered as portal resources at design time to be used at runtime.
- The `preferenceBean` provides the set of "default values" for the application, such as default navigation, catalog, page template, and skin. These values are defined in `adf-config.xml` and can be changed at runtime in the Preferences tab of the Portal Administration page.

```
<application>
    <default-render-kit-id>oracle.adf.rich</default-render-kit-id>
    <view-handler>
      oracle.webcenter.portalframework.sitestructure.handler.CustomViewHandler
    </view-handler>
</application>
<lifecycle>
    <phase-listener>oracle.webcenter.skin.view.SkinPhaseListener</phase-listener>
</lifecycle>
<managed-bean>
    <managed-bean-name>preferenceBean</managed-bean-name>
    <managed-bean-class>
      oracle.webcenter.portalframework.sitestructure.preference.PortalPreferences
    </managed-bean-class>
    <managed-bean-scope>application</managed-bean-scope>
</managed-bean>
```

H.5.4 Updating trinidad-config.xml

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > WEB-INF > trinidad-config.xml

Location on the filesystem:

- *PROJECT_ROOT*/public_html/WEB-INF/trinidad-config.xml

Detailed Instructions:

Add the following entry as a child of the root `<trinidad-config>` element, replacing any existing `<skin-family>` value. This EL expression allows the default skin to be changed at runtime; the value for the skin is provided by the

preferenceBean. The preferenceBean provides the set of "default values" for the application, such as default navigation, catalog, page template, and skin.

```
<skin-family>#{preferenceBean.defaultTrinidadSkin}</skin-family>
```

H.5.5 Updating web.xml

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > WEB-INF > web.xml

Location on the filesystem:

- *PROJECT_ROOT*/public_html/WEB-INF/web.xml

Detailed Instructions:

Add the following entries as children of the root <web-app> element. The <context-param> values should replace any existing values, as appropriate.

- The PortalAdminServlet redirects to the built in Portal Administration page. The link to this administration page is accessible at http://<server:port>/<context-root>/admin.
- The PortalFrameworkContextListener ensures that the application's memory references are properly cleaned up when it is deployed/undeployed.
- The JerseyWebApplicaton and restFacesContext entries configure the application to be able to use REST functionality.
- The parameter org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION is set to false by default for Framework applications for performance reasons, enabling compression for style class names.
- The parameter oracle.adf.view.rich.pprNavigation.OPTIONS is set to onWithForcePPR by default for Framework applications for performance reasons. This setting enables partial page navigation.

```
<context-param>
    <param-name>org.apache.myfaces.trinidad.DISABLE_CONTENT_COMPRESSION</param-name>
    <param-value>false</param-value>
  </context-param>
  <context-param>
    <param-name>oracle.adf.view.rich.pprNavigation.OPTIONS</param-name>
    <param-value>onWithForcePPR</param-value>
  </context-param>
  <listener>
    <description>Portal Framework Context Listener</description>
    <display-name>PortalFrameworkContextListener</display-name>
    <listener-class>
      oracle.webcenter.portalframework.sitestructure.internal.view.PortalFrameworkContextListener
    </listener-class>
  </listener>
  <servlet>
    <servlet-name>PortalAdminServlet</servlet-name>
    <servlet-class>oracle.webcenter.portalwebapp.servlet.PortalAdminServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>PortalAdminServlet</servlet-name>
    <url-pattern>/admin</url-pattern>
  </servlet-mapping>
  <servlet>
    <servlet-name>JerseyWebApplication</servlet-name>
```

```
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
    <init-param>
      <param-name>com.sun.jersey.config.property.packages</param-name>
      <param-value>
org.codehaus.jackson.jaxrs;oracle.webcenter.jaxrs;test.controller;oracle.webcenter.jaxrs.services.n
avigation.controller
      </param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.spi.container.ContainerRequestFilters</param-name>
      <param-value>
        oracle.webcenter.jaxrs.framework.filters.EmptyQueryParamRequestFilter
      </param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.spi.container.ContainerResponseFilters</param-name>
      <param-value>
oracle.webcenter.jaxrs.framework.filters.PaginatedLinkedFilter;oracle.webcenter.jaxrs.framework.fil
ters.HypermediaResponseFilter
      </param-value>
    </init-param>
    <init-param>
      <param-name>com.sun.jersey.config.property.resourceConfigClass</param-name>
      <param-value>com.sun.jersey.api.core.ClasspathResourceConfig</param-value>
    </init-param>
    <init-param>
      <param-name>token.names</param-name>
      <param-value>stoken</param-value>
    </init-param>
    <init-param>
      <param-name>token.manager.enabled</param-name>
      <param-value>false</param-value>
    </init-param>
  </servlet>
  <servlet-mapping>
    <servlet-name>JerseyWebApplication</servlet-name>
    <url-pattern>/api/*</url-pattern>
  </servlet-mapping>
  <filter>
    <filter-name>restFacesContext</filter-name>
    <filter-class>
      oracle.webcenter.jaxrs.services.sitestructure.faces.context.FacesContextFilter
    </filter-class>
  </filter>
  <filter-mapping>
    <filter-name>restFacesContext</filter-name>
    <servlet-name>JerseyWebApplication</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <filter-mapping>
    <filter-name>adfBindings</filter-name>
    <servlet-name>JerseyWebApplication</servlet-name>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <mime-mapping>
    <extension>html</extension>
    <mime-type>text/html</mime-type>
  </mime-mapping>
```

H.5.6 Updating adf-config.xml

Location in JDeveloper's Application Navigator:

- Application Resources > Descriptors > ADF META-INF > adf-config.xml

Location on the filesystem:

- *APPLICATION_ROOT*/.adf/META-INF/adf-config.xml

Detailed Instructions:

Add these entries as children of the root `<adf-config>` element. The highlighted (**bold faced**) `<namespace>` elements must be merged in with the existing `<mds-config>` entry, because this top-level entry should already exist.

- The `<mds-config>` element is the top-level element.
- The `<page-editor-config>` element sets up the default security policies for customization and personalization of pages and task flows within the Resource Manager. See also Chapter 17, "Adding and Using the Resource Manager."
- The `<adf-portal-config>` entry specifies the values of the default resources for the application. These values are the default values for the PreferenceBean and for the preferences that will appear in the Preferences tab of the WebCenter Portal: Spaces Administration console. Replace the highlighted paths with actual values (these default resources will be created in subsequent steps).
- The `<adf-rcs-config>` element defines the default resource catalog registry and navigation registry, which are used when editing catalogs and navigations in the Resource Manager. Replace the highlighted paths with actual values (these resources will be created in subsequent steps).
- The `<rcv-config>` element defines the default resource catalog, which is used when editing a page with Composer. This element includes the default resource catalog and the catalog selector. The catalog selector determines which catalog is used when editing a page. The `PortalCatalogSelector` attribute receives its values from the portal preference settings. The defaults for these settings are defined under `<adf-portal-config>`. The default catalog defined in `<rcv-config>` is only used if the catalog selector does not return a value. The preference setting takes precedence over the `<rcv-config>` setting. The highlighted path should be replaced with the path to the desired catalog. Any existing resource catalogs in the migrated application should be moved to the `/oracle/webcenter/portalapp` directory (created in subsequent steps), if they are to continue to be used in the Framework application. See also Chapter 16, "Creating and Managing Resource Catalogs."

```
<mdsC:adf-mds-config version="11.1.1.000">
   <mds-config xmlns="http://xmlns.oracle.com/mds/config">
     <persistence-config>
       <metadata-namespaces>
         ...
         <namespace path="/oracle/webcenter/portalapp"
                    metadata-store-usage="WebCenterFileMetadataStore"/>
         <namespace path="/oracle/webcenter/siteresources/scopedMD"
                    metadata-store-usage="WebCenterFileMetadataStore"/>
         <namespace path="/oracle/webcenter/security/scopedMD"
                    metadata-store-usage="WebCenterFileMetadataStore"/>
         <namespace path="/oracle/webcenter/siteresources/scopedMD/shared/"
                    metadata-store-usage="WebCenterFileMetadataStore"/>
```

```
          <namespace path="/oracle/webcenter/quicklinks/scopedMD"
                     metadata-store-usage="WebCenterFileMetadataStore"/>
          <namespace path="/oracle/webcenter/page/scopedMD"
                     metadata-store-usage="WebCenterFileMetadataStore"/>
          <namespace path="/pageDefs"
                     metadata-store-usage="WebCenterFileMetadataStore"/>
          ...
       </metadata-namespaces>
     </persistence-config>
   </mds-config>
 </mdsC:adf-mds-config>
<page-editor-config xmlns="http://xmlns.oracle.com/adf/pageeditor/config">
    <security-config>
      <security-policies>
        <security-policy name="customPagePolicy" override="pagePolicy">
          <policy-class>oracle.webcenter.security.policy.WCPageSecurityPolicy</policy-class>
        </security-policy>
        <security-policy name="customTFPolicy" override="taskFlowPolicy">
          <policy-class>oracle.webcenter.security.policy.WCTaskflowSecurityPolicy</policy-class>
        </security-policy>
      </security-policies>
    </security-config>
</page-editor-config>
<adf-portal-config xmlns="http://xmlns.oracle.com/webcenter/portal/config">
    <preferences>
      <preference id="oracle.webcenter.portalapp.navigation.model"
                        desc="Default Navigation Model"
                        value="/path/to/your/default/navigation.xml"
                        resourceType="navigation" display="true"/>
      <preference id="oracle.webcenter.portalapp.resourcecatalog"
                        desc="Default Resource Catalog"
                        value="/path/to/your/default/catalog.xml"
                        resourceType="ResourceCatalog" display="true"/>
      <preference id="oracle.webcenter.portalapp.pagetemplate.pageTemplate"
                        desc="Default Page Template"
                        value="/path/to/your/default/page/template.jspx"
                        resourceType="Template" display="true"/>
      <preference id="oracle.webcenter.portalapp.navigation.renderer"
                        desc="Default Navigation Renderer"
                        value="/oracle/webcenter/portalapp/pages/navigation-renderer.jspx"
                        display="false"/>
      <preference id="oracle.webcenter.portalapp.skin"
                        desc="Default Portal Skin" value="portal"
                        resourceType="Skin" display="true"/>
    </preferences>
  </adf-portal-config>
  <adf-rcs-config xmlns="http://xmlns.oracle.com/adf/rcs/adf-config">
    <rcs-config>
      <catalog-config default-scope="/"
                      default-registry="/path/to/your/default/catalog/registry.xml"/>
      <navigation-config default-scope="/"
                        default-registry="/path/to/your/default/navigation/registry.xml"/>
      <security-manager class-name=
"oracle.webcenter.portalframework.genericsiteresources.internal.security.CatalogSecurityManager"/>
    </rcs-config>
  </adf-rcs-config>
  <rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
    <default-catalog catalog-name="/path/to/your/default/catalog.xml"/>
    <catalog-selector
class-name="oracle.webcenter.portalframework.sitestructure.rc.PortalCatalogSelector"/>
```

```
</rcv-config>
```

H.6 Modifying the Project Configurations

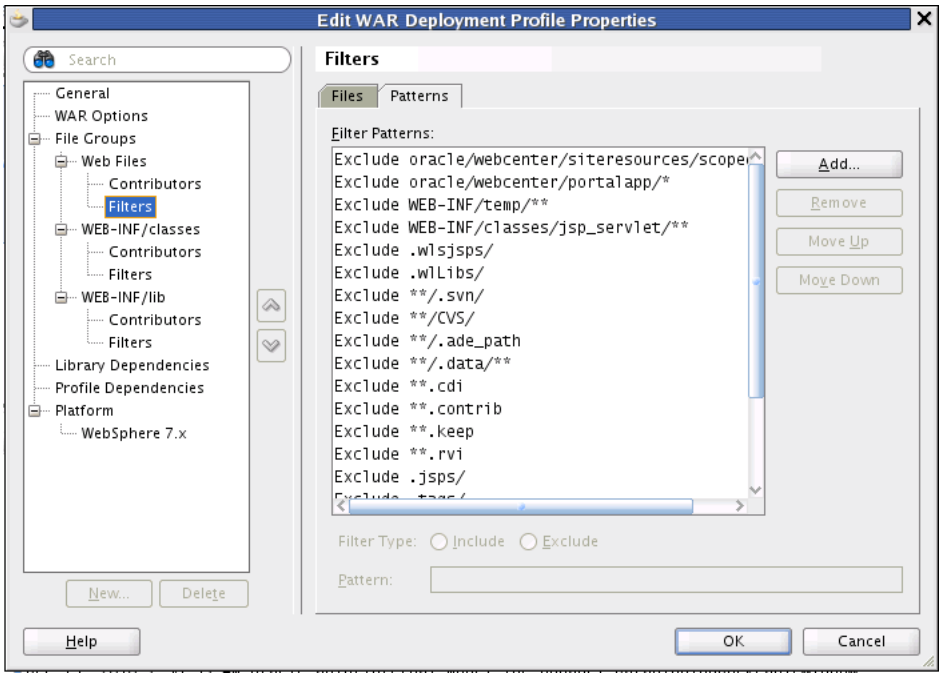
This section lists the project configuration files – JAR files, technology scopes, and WAR files – that must be added to the application.

- Add the following Oracle WebCenter Portal: Framework libraries to the project classpath. These libraries add the necessary Framework-specific JARs to the design time classpath. (To do this, right-click the project and chose **Project Properties** and then **Libraries and Classpaths**. Simply add the libraries from the list of available libraries.)
  - WebCenter Portal Framework
  - WebCenter Portal Framework REST API
- Add the following technology scope to the application. This technology scope identifies the application as a PS5 framework application. See also Section 6.1, "Manually Configuring a Framework Application With Technology Scopes."
  - WebCenter Portal – Framework
- Create a WAR deployment profile for project (or modify an existing one) and add pattern filters as described below. The deployment profile and filters ensure that the framework application's resources are properly packaged when it is deployed.
  - Web files
    - Exclude `oracle/webcenter/portalapp/*`
    - Exclude `oracle/webcenter/siteresources/scopedMD/*`
  - WEB-INF/classes
    - Exclude `oracle/webcenter/portalapp/*/*.dcx`
    - Exclude `oracle/webcenter/portalapp/*/*.cpx`
    - Exclude `oracle/webcenter/portalapp/*/*.xml`

  The basic procedure for adding the filters is:

  1. Right-click the project and select **Project Properties**.
  2. In the Project Properties dialog, select the **Deployment** node.
  3. In the Deployment Profiles list, select the WAR file deployment profile for your framework application and click **Edit**.
  4. In the Edit WAR Deployment Profile Properties dialog, open the **File Groups** node and select **Filters** under the **Web Files** node.
  5. Select the **Patterns** tab, as shown in Figure H–1.

Figure H–1 Edit WAR Deployment Profile Properties Dialog



6. Select the file patterns to exclude.

The steps for excluding `WEB-INF/classes` is similar the above procedure. Instead of selecting **Web Files** and then **Filters**, select **WEB-INF/classes** and then **Filters**.

H.7 Creating Portal-Specific Files

A Framework application requires a specific file structure. This section explains how to create and configure this file structure. See also Section 9.12, "How are WebCenter Portal Files Organized?."

- Section H.7.1, "Create the Framework Application Folder Structure"
- Section H.7.2, "Create a Resource Registry File"
- Section H.7.3, "Create Default Resources"

H.7.1 Create the Framework Application Folder Structure

Create the portal folder structure in the specified location.

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content

Location on the filesystem:

- *PROJECT_ROOT*/public_html

Detailed Instructions:

Create the following folder structure in the location specified above (that is, in the Web Content folder in JDev). To add a folder, right-click the `portalapp` folder and select **General** in the New Gallery dialog. Then, select **Folder** from the list of Items.

- `oracle/webcenter/portalapp/`

- `catalogs/`
- `navigations/`
- `pages/`
- `pagetemplates/`
- `shared/`

The `/oracle/webcenter/portalapp` folder structure is required, and all portal resources must be created under this namespace. For more information on the file structure, see Section 9.12.1.2, "Why Is a WebCenter Portal: Framework Application Organized the Way It Is?."

The additional subfolders (except for the shared folder) are not strictly required, but are useful for organizational purposes. However, the `shared` folder must exist. This folder provides the path under which any resources associated with a portal resource (such as images) must be located.

H.7.2 Create a Resource Registry File

The `generic-site-resources.xml` file is required for resources to be registered with the runtime Resource Manager. For more information on the Resource Manager, see Chapter 17, "Adding and Using the Resource Manager."

Create a resource registry file under `APPLICATION_ROOT/mds/oracle/webcenter/siteresources/scopedMD/s8bba98ff_4cbb_40b8_beee_296c916a23ed/generic-site-resources.xml` with the following content. Note that you must create this folder structure manually; it does not exist by default.

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<resources xmlns="http://xmlns.oracle.com/webcenter/portalframework/genericSiteResources"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <resourceType name="siteTemplate" resourceBundle=""/>
  <resourceType name="pageStyle" resourceBundle=""/>
  <resourceType name="dataPresenter" resourceBundle=""/>
  <resourceType name="contentPresenter" resourceBundle=""/>
  <resourceType name="resourceCatalog" resourceBundle=""/>
  <resourceType name="navigation" resourceBundle=""/>
  <resourceType name="skin" resourceBundle=""/>
  <resourceType name="taskFlowStyle" resourceBundle=""/>
  <resourceType name="taskFlow" resourceBundle=""/>
  <resourceType name="dataControl" resourceBundle=""/>
</resources>
```

H.7.3 Create Default Resources

This section explains how to create and register several portal resource files.

H.7.3.1 Default Navigation Model File

The navigation model file, `default-navigation-model.xml`, allows you to incorporate navigation structures in portal site. For more information, see Chapter 11, "Building a Navigation Model for Your Portal" and Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/navigations

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/navigations

Procedure:

1. Right-click `/oracle/webcenter/portalapp/navigations` folder in Application Navigator and select **New**.
2. In the New Gallery dialog, open the **Portal** node under **Web Tier** and select **Navigation**.
3. Enter a name (for example, `default-navigation-model.xml`), ensure directory is PROJECT_ROOT`/public_html/oracle/webcenter/portalapp/navigations`, and check the box labeled **Create as a Portal Resource**.

H.7.3.2 Navigation Registry File

The navigation registry file, `navigation-registry.xml`, is used for editing navigations in the Resource Manager. See also Chapter 11, "Building a Navigation Model for Your Portal" and Section 9.5.3, "Understanding the Navigation Model and the Navigation Registry."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/navigations

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/navigations

Procedure:

1. Create a new Framework application. You will copy some files from this new application into the application you are migrating.
2. From the new application, copy this file:

   ```
   <PROJECT_ROOT>/public_
   html/oracle/webcenter/portalapp/navigations/navigation-registry.xml
   ```

   to this location in the application you are migrating:

   ```
   <PROJECT_ROOT>/oracle/webcenter/portalapp/navigations/navigation-registry.xml
   ```

3. Add two top-level folders, called `new` and `links`.
4. Under the `new` folder, add the set of resources that should allowed to be added to a navigation when editing at runtime.
5. Under the `links` folder, add the set of resources that should appear in the links picker, which is displayed when adding an item of type "Link" to a navigation at runtime. For more information, see Chapter 11, "Building a Navigation Model for Your Portal."

H.7.3.3 Resource Catalog File

A catalog specifies a collection of an otherwise unrelated group of elements, like layout components, task flows, portlets, documents, and others, that an authorized user can add to a portal at runtime. Oracle WebCenter Portal's Composer uses catalogs at runtime to determine which elements an authorized user can add to a portal page. See also Chapter 17, "Adding and Using the Resource Manager" and Section 9.5.4, "Understanding Resource Catalogs and the Catalog Registry."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/catalogs

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/catalogs

Follow one of the following two procedures. The first explains how to create a new resource catalog, and the second explains how to reuse resource catalogs from your current application.

Procedure:

1. Right-click `/oracle/webcenter/portalapp/catalogs` folder in Application Navigator and select **New**.
2. In the New Gallery dialog, open the **Portal** node under **Web Tier** and select **Application Resource Catalog**.
3. Enter a name (for example, `default-catalog.xml`), ensure directory is PROJECT_ROOT`/public_html/oracle/webcenter/portalapp/catalogs`, and check the box labeled **Create as a Portal Resource**.
4. Add the set of resources that should allowed to be added to a page when editing at runtime. For more information, see Section 17.2, "Working with Portal Resources in JDeveloper."

Alternate Procedure:

Consider following this procedure if you have existing resource catalogs that you would like to reuse:

1. Copy any existing resource catalogs your application may have (normally found in `APPLICATION_ROOT/mds/oracle/adf/rc/metadata`) to `PROJECT_ROOT/public_html/oracle/webcenter/portalapp/catalogs`.
2. Right-click on each catalog under `PROJECT_ROOT/public_html/oracle/webcenter/portalapp/catalogs` and choose **Create Portal Resource**.

   This alternate procedure registers each catalog as a portal resource so it appears in the Resources tab of the WebCenter Portal Administration Console.

H.7.3.4 Resource Catalog Registry File

The resource catalog registry file defines the set of items that are available for inclusion in your resource catalog at runtime. See Chapter 16, "Creating and Managing Resource Catalogs."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/catalogs

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/catalogs

Procedure:

1. Create a new Framework application. You will copy some files from this new application into the application you are migrating.
2. From the new application, copy this file:

   `<PROJECT_ROOT>/public_`

```
html/oracle/webcenter/portalapp/catalogs/catalog-registry.xml
```

to this location in the application you are migrating:

```
<PROJECT_ROOT>/oracle/webcenter/portalapp/catalogs/catalog-registry.xml
```

H.7.3.5 Page Template File

Page templates allow you to define entire page layouts and apply them to pages to create a consistent layout across the portal. For more information, see Chapter 12, "Designing the Look and Feel of Your Portal."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/pagetemplates

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/pagetemplates

Procedure:

1. Right-click the `/oracle/webcenter/portalapp/pagetemplates` folder in Application Navigator and select **New**.
2. In the New Gallery dialog, open the **JSF** node under **Web Tier** and select **JSF Page Template**.
3. Check the box labeled **Create Associated ADFm Page Definition** and add a facet definition named `content`.
4. Add a reference to the `content` facet within the page template once it has been created. For example:

```
<?xml version='1.0' encoding='UTF-8'?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
          xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
  <jsp:directive.page contentType="text/html;charset=UTF-8"/>
  <af:pageTemplateDef var="attrs">
    <af:xmlContent>
      <component xmlns="http://xmlns.oracle.com/adf/faces/rich/component">
        <display-name>Default Page Template</display-name>
        <facet>
          <facet-name>content</facet-name>
        </facet>
      </component>
    </af:xmlContent>
    <af:facetRef facetName="content"/>
  </af:pageTemplateDef>
</jsp:root>
```

5. Right-click the page template in the Application Navigator and select **Create Portal Resource**, then click **OK**.

H.7.3.6 Navigation Renderer File

The navigation renderer file is responsible for rendering non-page resources that appear in a navigation, such as portlets, content, and task flows. For more information, see Section 11.5, "Editing the Navigation Renderer."

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/pages/

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/pages/

Procedure:

1. In the folder specified by the location listed above, create a new JSF page. From the New menu, select **Web Tier**, then **JSF**, then **JSF Page**. Name the file `navigation-renderer.jspx`.
2. Replace the file content with the following:

```
<?xml version='1.0' encoding='UTF-8'?>
 <jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
           xmlns:f="http://java.sun.com/jsf/core"
           xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
   <jsp:directive.page contentType="text/html;charset=UTF-8"/>
   <f:view>
     <af:document id="d1"

title="#{SiteStructureContext.currentNavigationModel.currentSelection.title}">
       <af:form id="f1">
         <af:pageTemplate
value="#{bindings.pageTemplateBinding.templateModel}"
                          id="pt1">
           <f:facet name="content">
             <af:region value="#{bindings.sitestructurerenderer1.regionModel}"
                        id="r1"/>
           </f:facet>
         </af:pageTemplate>
       </af:form>
     </af:document>
   </f:view>
 </jsp:root>
```

H.7.3.7 Page Hierarchy File

The `pages.xml`, or "page hierarchy" file, organizes pages into a tree structure, with a parent-child relationship between pages. This hierarchical structure allows the convenient propagation or inheritance of security settings from pages to sub pages. See Section 9.5.1, "Understanding Pages, Page Templates, and the Portal Page Hierarchy."

> **Note:** After creating the `pages.xml` file in JDeveloper, JDeveloper must be restarted in order for the `pages.xml` to be recognized as a page hierarchy file and for the hierarchy editor to be displayed when it is opened.

Location in JDeveloper's Application Navigator:

- *PROJECT_NAME* > Web Content > oracle/webcenter/portalapp/pagehierarchy

Location on the filesystem:

- *PROJECT_ROOT*/public_html/oracle/webcenter/portalapp/pagehierarchy

Procedure:

1. In the folder specified by the location listed above, create a new file called `pages.xml`.

2. Add the following content to the file:

```
<?xml version="1.0" encoding="US-ASCII" ?>
   <pagesDef xmlns="http://xmlns.oracle.com/webcenter/page"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation=
          "http://xmlns.oracle.com/webcenter/page/pagesDef.xsd"
  />
```

H.8 Applying Security Configurations

This section explains how to configure security in the Framework application.

- Section H.8.1, "Configure ADF Security"
- Section H.8.2, "Configure the jazn-data.xml File"

H.8.1 Configure ADF Security

Ensure ADF Security is configured for ADF Authentication and Authorization. For more information, see Section 69.3.1, "Configuring ADF Security Settings." If security was not already configured for the application, you can configure it as follows:

1. From the Application menu, select **Secure** and then **Configure ADF Security**.
2. Select the **ADF Authentication and Authorization** option.
3. Click **Finish**.

H.8.2 Configure the jazn-data.xml File

These permissions are required for the proper functioning of the Portal Administration page and the Resource Manager at runtime.

> **Note:** Do not replace the contents of `jazn-data.xml` with the following contents; rather, merge the highlighted (**bold faced**) elements with the existing `<grant>` and `<permissions>` entries.

1. In the Application Navigator, open the **Application Resources** folder, then open **Descriptors**, and then the **META-INF** folder.
2. In the **META-INF** folder, open the `jazn-data.xml` file.
3. Add an Application Role called **Administrator**, if one doesn't exist. See Section 69.2, "Creating an Application Role."
4. Manually insert permission entries into the source file as follows. Again, be careful to *merge the highlighted elements* rather than replace the entire contents of the file with the following code.

```
<grant>
    <grantee>
      <principals>
        <principal>
          <class>oracle.security.jps.internal.core.principals.JpsAuthenticatedRoleImpl</class>
          <name>authenticated-role</name>
        </principal>
      </principals>
    </grantee>
```

```
    <permissions>
      ...
      <permission>
         <class>oracle.adf.share.security.authorization.RegionPermission</class>
         <name>oracle.webcenter.portalwebapp.pages.adminPageDef</name>
         <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.taskflow.view.ContainerPageDef</name>
        <actions>grant</actions>
      </permission>
      <permission>
        <class>oracle.webcenter.security.auth.HierarchicalResourcePermission</class>
        <name>serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_
296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed</name>
        <actions>personalize,view</actions>
      </permission>
      <permission>
        <class>oracle.adf.rc.security.CatalogPermission</class>
        <name>/oracle/webcenter/quicklinks/scopedMD/.*</name>
        <actions>create,edit</actions>
      </permission>
      ...
    </permissions>
  </grant>
  <grant>
    <grantee>
      <principals>
        <principal>
          <class>oracle.security.jps.internal.core.principals.JpsAnonymousRoleImpl</class>
          <name>anonymous-role</name>
        </principal>
      </principals>
    </grantee>
    <permissions>
      ...
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.siteresources.scopedMD.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.controller.security.TaskFlowPermission</class>
        <name>/oracle/webcenter/siteresources/scopedMD/.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.controller.security.TaskFlowPermission</class>
        <name>/oracle/webcenter/portalapp/.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.taskflow.view.ContainerPageDef</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.taskflow.view.ViewerPageDef</name>
```

```
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.taskflowstyle.view.PreviewPageDef</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.page.pstemplates.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.share.security.authorization.RegionPermission</class>
        <name>oracle.webcenter.portalapp.pagetemplates.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.webcenter.security.auth.HierarchicalResourcePermission</class>
        <name>serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_
296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.controller.security.TaskFlowPermission</class>
        <name>/oracle/webcenter/navigationtaskflows/view/.*</name>
        <actions>view</actions>
      </permission>
      <permission>
        <class>oracle.adf.controller.security.TaskFlowPermission</class>
        <name>/oracle/webcenter/.*</name>
        <actions>view</actions>
      </permission>
      ...
    </permissions>
  </grant>
  <grant>
    <grantee>
      <principals>
        <principal>
          <class>oracle.security.jps.service.policystore.ApplicationRole</class>
          <name>Administrator</name>
          <guid>843A31B1C1B211DFBFA9CF5E373CA556</guid>
        </principal>
      </principals>
    </grantee>
    <permissions>
      ...
       <permission>
           <class>oracle.webcenter.security.auth.HierarchicalResourcePermission</class>
           <name>serviceID=oracle.webcenter.page,scopeID=s8bba98ff_4cbb_40b8_beee_
296c916a23ed,resourceID=s8bba98ff_4cbb_40b8_beee_296c916a23ed</name>
           <actions>create,delete,grant,manage,personalize,update,view</actions>
       </permission>
       <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_
contentpresenter_.*</name>
          <actions>manage</actions>
     </permission>
```

```
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_datacontrol_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_navigation_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_pagestyle_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_catalog_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_pagetemplate_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_skin_.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_taskflow_
.*</name>
          <actions>manage</actions>
     </permission>
     <permission>
          <class>oracle.webcenter.security.auth.WebCenterResourcePermission</class>
          <name>oracle_webcenter_siteresource_s8bba98ff_4cbb_40b8_beee_296c916a23ed_taskflowstyle_
.*</name>
          <actions>manage</actions>
     </permission>
      ...
  </permissions>
</grant>
```

WebCenter Portal Analytics Database Schema

This appendix describes the underlying database design for Oracle WebCenter Portal's Analytics. This information is useful to developers building *custom* analytics data queries, as described in Section 47.3, "Building Analytics Reports.".

This appendix contains the following topics:

- Section I.1, "Analytics Database Schema Overview"
- Section I.2, "Analytics Fact Table Descriptions"
- Section I.3, "Analytics Dimension Table Descriptions"
- Section I.4, "Analytics System Table Descriptions"
- Section I.5, "Analytics User Properties"
- Section I.6, "Analytics Event and Dimension Data Types"

I.1 Analytics Database Schema Overview

This section contains the following topics:

- Section I.1.1, "Analytics Database Model Overview"
- Section I.1.2, "Analytics Database Table Overview"

I.1.1 Analytics Database Model Overview

Oracle WebCenter Portal's Analytics database schema is modeled as a star-schema to optimize performance and provide fast response times.

Analytics data is stored in fact and dimension fact tables:

- **Analytics Fact Tables** are created when the Analytics Collector runs for the first time. Each fact table stores a specific event type, for example, page views, space views, document uploads, blog views, wiki edits, and so on. Columns in these fact tables mostly contain integer IDs that reference descriptive data stored in dimension tables.

  Table I–1, " Analytics Facts Tables (Prefixed ASFACT_WC)" lists types of analytics events that can be collected and stored, and the associated tables.

- **Analytics Dimension Tables** are also created when the Analytics Collector runs for the first time. Dimension tables store redundant descriptive data associated with analytics events. When a value is not found in these tables, a new record is

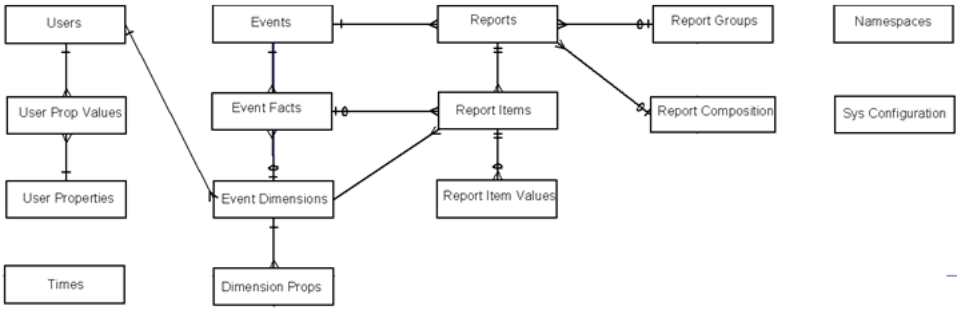
added. Typical analytics dimension data includes: page name, browser name, document name, space name, and so on.

Table I–2, " Analytics Dimension Tables (Prefixed ASDIM_WC)" lists the different types of analytics dimensions that can be collected and stored, and the associated table names.

The Analytics database schema also contains a few application tables that store information such as system configuration parameters. Application tables are created during installation.

Figure I–1 shows the relationship between predefined fact, dimension, and application tables in the analytics database schema.

Figure I–1 Analytics Database Model



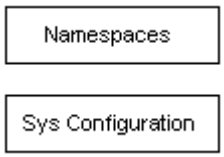
Tables shown in Figure I–1 store the following analytics data:

- Application configuration
- Event metadata
- Report metadata

Application Configuration

Figure I–2 shows which entities are used to store system parameters for the Analytics service.

Figure I–2 Application Configuration Entities

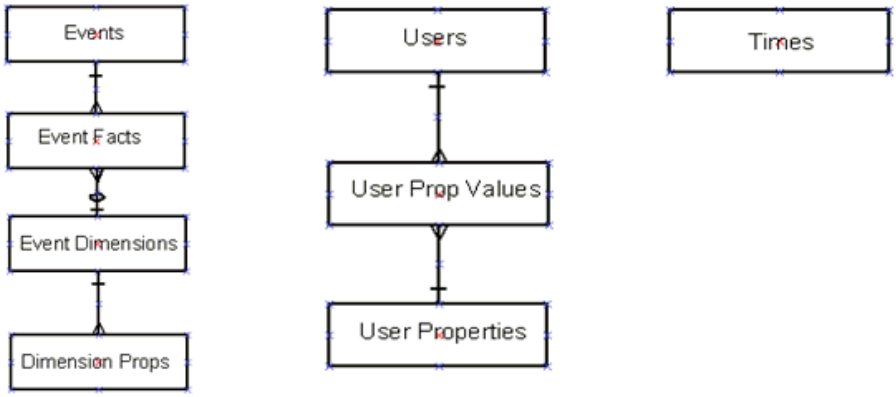


Event Metadata

Figure I–3 shows which entities are used to store event data. The Users and Times entities are mainly dimensions that provide user/time oriented views of the event data.

Event metadata records which events are collected, the tables where events are stored, as well as dimensions that are saved each time an event occurs.

Figure I–3 Event Metadata Entities

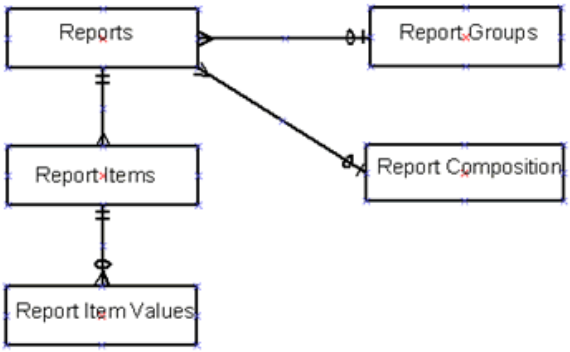


Report Metadata

Figure I–4 shows which entities represent out-of-the-box analytics reports. Metadata in these tables are the primary source for information such as report metrics, report structure, and columns listed.

Report metadata stores out-of-the-box report configurations for WebCenter Portal's Analytics. Each report contains many columns to display in the report and which filters to apply. Some reports are composed from other reports.

Figure I–4 Report Metadata Entities



I.1.2 Analytics Database Table Overview

This section lists and describes database tables in the analytics schema:

- Table I–1, " Analytics Facts Tables (Prefixed ASFACT_WC)"
- Table I–2, " Analytics Dimension Tables (Prefixed ASDIM_WC)"
- Table I–3, " Analytics System Tables (Prefixed ASSYS)"

Note: Table names for analytics event and dimension tables are automatically generated based on analytics event and dimension names. The mapping between analytics event and dimension names and their corresponding table names is defined in the ASSYS_EVENTS and ASSYS_EVENTDIMENSIONS tables.

For an example of how to use a query to return the table names associated with a specific event, see Section I.1.2.1, "Sample SQL Query: Finding Table Names Associated with Specific Events".

Table I–1 Analytics Facts Tables (Prefixed ASFACT_WC)

Event Object	Table Name	Event Table Description
Pages	ASFACT_WC_PAGECRE_0	Stores event information captured by the Analytics service when a page is created.
	ASFACT_WC_PAGEDEL_0	Stores event information captured by the Analytics service when a page is deleted.
	ASFACT_WC_PAGEEDI_0	Stores event information captured by the Analytics service when a page is edited.
	ASFACT_WC_PAGETAG_0	Stores event information captured by the Analytics service when a page is tagged.
	ASFACT_WC_PAGEVIE_0	Stores event information captured by the Analytics service when a page is viewed.
Discussions	ASFACT_WC_DISCUSS_0	Stores event information captured by the Analytics service when a discussion topic is viewed.
	ASFACT_WC_DISCUSS_1	Stores event information captured by the Analytics service when an announcement is viewed.
	ASFACT_WC_DISCUSS_2	Stores event information captured by the Analytics service when a discussion forum is created.
	ASFACT_WC_DISCUSS_3	Stores event information captured by the Analytics service when a discussion forum is deleted.
	ASFACT_WC_DISCUSS_4	Stores event information captured by the Analytics service when a discussion topic is created.
	ASFACT_WC_DISCUSS_5	Stores event information captured by the Analytics service when a discussion topic is edited.
	ASFACT_WC_DISCUSS_6	Stores event information captured by the Analytics service when a discussion topic is deleted.
	ASFACT_WC_DISCUSS_7	Stores event information captured by the Analytics service when a discussion topic is tagged.
	ASFACT_WC_DISCUSS_8	Stores event information captured by the Analytics service when someone replies to a discussion topic message.
	ASFACT_WC_DISCUSS_9	Stores event information captured by the Analytics service when a discussion topic message is liked.
	ASFACT_WC_DISCUSS_A	Stores event information captured by the Analytics service when an announcement is created.
	ASFACT_WC_DISCUSS_B	Stores event information captured by the Analytics service when an announcement is edited.
	ASFACT_WC_DISCUSS_C	Stores event information captured by the Analytics service when an announcement is deleted.

Table I–1 (Cont.) Analytics Facts Tables (Prefixed ASFACT_WC)

Event Object	Table Name	Event Table Description
Documents - Blog - Wikis	ASFACT_WC_DOCLIB__0	Stores event information captured by the Analytics service when a document is viewed.
	ASFACT_WC_DOCLIB__1	Stores event information captured by the Analytics service when a document is downloaded.
	ASFACT_WC_DOCLIB__2	Stores event information captured by the Analytics service when a document is created.
	ASFACT_WC_DOCLIB__3	Stores event information captured by the Analytics service when a document is edited.
	ASFACT_WC_DOCLIB__4	Stores event information captured by the Analytics service when a document is tagged.
	ASFACT_WC_DOCLIB__5	Stores event information captured by the Analytics service when a document is liked.
	ASFACT_WC_DOCLIB__6	Stores event information captured by the Analytics service when a document is commented.
	ASFACT_WC_DOCLIB__7	Stores event information captured by the Analytics service when a document is deleted.
Space Events	ASFACT_WC_EVENT_C_0	Stores event information captured by the Analytics service when a space event is created.
	ASFACT_WC_EVENT_D_0	Stores event information captured by the Analytics service when a space event is edited.
	ASFACT_WC_EVENT_E_0	Stores event information captured by the Analytics service when a space event is deleted.
Spaces	ASFACT_WC_GROUPSP_0	Stores event information captured by the Analytics service when a space is viewed.
	ASFACT_WC_GROUPSP_1	Stores event information captured by the Analytics service when a space is created.
	ASFACT_WC_GROUPSP_2	Stores event information captured by the Analytics service when someone joins a space.
	ASFACT_WC_GROUPSP_3	Stores event information captured by the Analytics service when a space is deleted.
Lists	ASFACT_WC_LIST_CR_0	Stores event information captured by the Analytics service when a list is created.
	ASFACT_WC_LIST_DE_0	Stores event information captured by the Analytics service when a list is deleted.
	ASFACT_WC_LIST_ED_0	Stores event information captured by the Analytics service when a list is edited.
Logins	ASFACT_WC_LOGINS_0	Stores event information captured by the Analytics service when a user logs in to the site.
People	ASFACT_WC_PEOPLEC_0	Stores event information captured by the Analytics service when someone adds a connection.
	ASFACT_WC_PEOPLEC_1	Stores event information captured by the Analytics service when someone posts on a wall.
	ASFACT_WC_PEOPLEC_2	Stores event information captured by the Analytics service when someone edits a profile.
	ASFACT_WC_PEOPLEC_3	Stores event information captured by the Analytics service when someone edits status information.

Table I–1 (Cont.) Analytics Facts Tables (Prefixed ASFACT_WC)

Event Object	Table Name	Event Table Description
Portlets	ASFACT_WC_PORTLET_0	Stores event information captured by the Analytics service when a portlet is viewed.
Searches	ASFACT_WC_SEARCHE_0	Stores event information captured by the Analytics service when a search is performed.

Table I–2 Analytics Dimension Tables (Prefixed ASDIM_WC)

Dimension Name	Table Name	Dimension Table Description
Applications	ASDIM_WC_APPLICA_0	Stores names of WebCenter Portal applications accessing analytics data.
IP Clients	ASDIM_WC_CLIENT__0	Stores client IP addresses received in analytics events.
Discussion Topics	ASDIM_WC_DISCUSS_0	Stores discussion topic details received in analytics events.
Discussion Forums	ASDIM_WC_DISCUSS_1	Stores discussion forum details received in analytics events.
Discussion Messages	ASDIM_WC_DISCUSS_2	Stores discussion message details received in analytics events.
Announcements	ASDIM_WC_DISCUSS_3	Stores announcement details received in analytics events.
Documents	ASDIM_WC_ DOCUMENT_0	Stores document details received in analytics events.
Lists	ASDIM_WC_LISTS_0	Stores list details received in analytics events.
Pages	ASDIM_WC_PAGES_0	Stores WebCenter Portal page details received in analytics events.
Portlets	ASDIM_WC_PORTLET_0	Stores portlet information received in analytics events.
Portlet Instances	ASDIM_WC_PORTLET_1	Stores portlet instance information received in analytics events.
Portlet Producers	ASDIM_WC_PRODUCE_0	Stores producers associated with portlet.
Referrers	ASDIM_WC_REFERRE_0	Stores referred URLs where analytics events come from.
Searches	ASDIM_WC_SEARCHE_0	Stores search phrases received in analytics events.
Space Events	ASDIM_WC_EVENTS_0	Stores space event details, received in analytics events.
Spaces	ASDIM_WC_GROUPSP_0	Stores space information received in analytics events.
Tags	ASDIM_WC_TAGS_0	Stores tags received in analytics events.
Browsers	ASDIM_WC_USER_AG_0	Stores information about which browsers are used to access analytics.
Times	ASDIM_TIME	Stores times used to group analytics reports.
Users	ASDIM_USERS	Stores user details received in analytics events.
User Properties	ASDIM_ USERPROPERTIES	Stores user properties.

Table I–2 (Cont.) Analytics Dimension Tables (Prefixed ASDIM_WC)

Dimension Name	Table Name	Dimension Table Description
User Property Values	ASDIM_ USERPROPERTYVALUES	Stores user property values.

Table I–3 Analytics System Tables (Prefixed ASSYS)

Entity Name	Table Name	Table Description
Configuration	ASSYS_CONFIG	Stores system configuration details.
Dimension Properties	ASSYS_ DIMENSIONPROPS	Stores configuration details for each dimension.
Event Dimensions	ASSYS_ EVENTDIMENSIONS	Stores dimensions associated with each event.
Event Facts	ASSYS_EVENTFACTS	Stores facts associated with each event.
Events	ASSYS_EVENTS	Stores the different types of analytics events that are collected by the Analytics service. An analytics event represents a single user action, such as document view, user login, and so on.
Namespaces	ASSYS_NAMESPACES	Stores the namespace in which analytics events are registered.
Report Composition	ASSYS_ REPORTCOMPOSITION	Stores out-of-the-box composite analytics reports.
Report Group	ASSYS_REPORTGROUP	Stores out-of-the-box analytics report groups, such as portlets, services, and so on.
Report Items	ASSYS_REPORTITEMS	Stores items included in each analytics report.
Report Item Values	ASSYS_ REPORTITEMVALUES	Stores values associated with each item in an analytics report.
Reports	ASSYS_REPORTS	Stores out-of-the-box analytics reports.

I.1.2.1 Sample SQL Query: Finding Table Names Associated with Specific Events

This section provides a sample SQL query to return the names of tables associated with a specific event. This query uses a filter to return tables associated with the Documents service.

```
SELECT tablename "Table Name",
       displayname "Event Display Name"
FROM   assys_events
WHERE tablename like 'ASFACT_WC_DOCLIB%'
ORDER BY 1;
```

Sample Report Output

```
Table Name            Event Display Name
----------------------------------------
ASFACT_WC_DOCLIB__0   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTVIEWS
ASFACT_WC_DOCLIB__1   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTDOWNLOADS
ASFACT_WC_DOCLIB__2   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTCREATE
ASFACT_WC_DOCLIB__3   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTEDIT
ASFACT_WC_DOCLIB__4   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTTAG
ASFACT_WC_DOCLIB__5   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTLIKE
ASFACT_WC_DOCLIB__6   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTCOMMENT
ASFACT_WC_DOCLIB__7   {HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DOCLIB_DOCUMENTDELETES
```

I.2 Analytics Fact Table Descriptions

Analytics fact tables are created when the Analytics Collector runs for the first time. Each fact table stores a specific event type, for example, page views, space views, document uploads, blog views, wiki edits, and so on. Columns in these fact table mostly contain integer IDs that reference descriptive data stored in dimension tables.

This section describes each of the tables listed in Table I–1, " Analytics Facts Tables (Prefixed ASFACT_WC)".

I.2.1 ASFACT_WC_PAGECRE_0

Fact table that stores event information captured by the Analytics service when someone creates a page.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time the dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID.	ASDIM_WC_ APPLICA_0.ID

I.2.2 ASFACT_WC_PAGEDEL_0

Fact table that stores event information captured by the Analytics service when someone deletes a page.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time the dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.3 ASFACT_WC_PAGEEDI_0

Fact table that stores event information captured by the Analytics service when someone edits a page.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID

I.2.4 ASFACT_WC_PAGETAG_0

Fact table that stores event information captured by the Analytics service when someone tags a page.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
TAG_	NULL	NUMBER(38)	Tag ID dimension.	ASDIM_WC_USER_ AG_0.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.5 ASFACT_WC_PAGEVIE_0

Fact table that stores event information captured by the Analytics service when someone views a page.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
RESPONSE_ TIME_	NULL	NUMBER(16)	Response time (ms).	
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID.	
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_ REFERRE_0.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_USER_ AG_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.6 ASFACT_WC_DISCUSS_0

Fact table that stores event information captured by the Analytics service when someone views a discussion topic.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_USER_ AG_0.ID
FORUM_	NULL	NUMBER(38)	Discussion forum associated with the event.	ASDIM_WC_ DISCUSS_1.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID.	
TOPIC_	NULL	NUMBER(38)	Discussion topic associated with the event.	ASDIM_WC_DISCUSS_0.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_REFERRE_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID

I.2.7 ASFACT_WC_DISCUSS_1

Fact table that stores event information captured by the Analytics service when someone views an announcement.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_EVENTFACTS.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_CLIENT__0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_REFERRE_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_USER_AG_0.ID
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID.	
ANNOUNCEMENT_	NULL	NUMBER(38)	Announcement associated with the event.	ASDIM_WC_DISCUSS_3.ID

I.2.8 ASFACT_WC_DISCUSS_2

Fact table that stores event information captured by the Analytics service when someone creates a discussion forum.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
DISCUSSION_ FORUM_	NULL	NUMBER(38)	Discussion forum ID dimension.	ASDIM_WC_ DISCUSS_0.ID

I.2.9 ASFACT_WC_DISCUSS_3

Fact table that stores event information captured by the Analytics service when someone deletes a discussion forum.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
DISCUSSION_ FORUM_	NULL	NUMBER(38)	Discussion forum ID dimension.	ASDIM_WC_ DISCUSS_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.10 ASFACT_WC_DISCUSS_4

Fact table that stores event information captured by the Analytics service when someone creates a discussion topic.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
DISCUSSION_ TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_ DISCUSS_0.ID
DISCUSSION_ MESSAGE_	NULL	NUMBER(38)	Discussion message ID dimension.	ASDIM_WC_ DISCUSS_2.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID

I.2.11 ASFACT_WC_DISCUSS_5

Fact table that stores event information captured by the Analytics service when someone edits a discussion topic.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID
DISCUSSION_TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_DISCUSS_0.ID

I.2.12 ASFACT_WC_DISCUSS_6

Fact table that stores event information captured by the Analytics service when someone deletes a discussion topic.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
DISCUSSION_TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_DISCUSS_0.ID

I.2.13 ASFACT_WC_DISCUSS_7

Fact table that stores event information captured by the Analytics service when someone tags a discussion topic.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
TAG_	NULL	NUMBER(38)	Tag ID dimension.	ASDIM_WC_ TAGS_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
DISCUSSION_ TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_ DISCUSS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.14 ASFACT_WC_DISCUSS_8

Fact table that stores event information captured by the Analytics service when someone replies to a discussion topic message.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
DISCUSSION_ MESSAGE_	NULL	NUMBER(38)	Discussion message ID dimension.	ASDIM_WC_ DISCUSS_2.ID
DISCUSSION_ TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_ DISCUSS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.15 ASFACT_WC_DISCUSS_9

Fact table that stores event information captured by the Analytics service when someone likes a discussion topic message.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
DISCUSSION_ MESSAGE_	NULL	NUMBER(38)	Discussion message ID dimension.	ASDIM_WC_ DISCUSS_2.ID
APPLICATION _	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
DISCUSSION_ TOPIC_	NULL	NUMBER(38)	Discussion topic ID dimension.	ASDIM_WC_ DISCUSS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.16 ASFACT_WC_DISCUSS_A

Fact table that stores event information captured by the Analytics service when someone creates an announcement.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
DISCUSSION_ ANNOUNCEM ENT_	NULL	NUMBER(38)	Announcement ID dimension.	ASDIM_WC_ DISCUSS_3.ID

I.2.17 ASFACT_WC_DISCUSS_B

Fact table that stores event information captured by the Analytics service when someone edits an announcement.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
DISCUSSION_ ANNOUNCEM ENT_	NULL	NUMBER(38)	Announcement ID dimension.	ASDIM_WC_ DISCUSS_3.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.18 ASFACT_WC_DISCUSS_C

Fact table that stores event information captured by the Analytics service when someone deletes an announcement.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
DISCUSSION_ FORUM_	NULL	NUMBER(38)	Discussion forum ID dimension.	ASDIM_WC_ DISCUSS_1.ID

I.2.19 ASFACT_WC_DOCLIB__0

Fact table that stores event information captured by the Analytics service when someone views a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	DATE	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
RESPONSE_ TIME_	NULL	NUMBER(16)	Response Time (ms).	
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID.	
GROUPSPACE	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
REFERRER	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_ REFERRE_0.ID
DOCUMENT	NULL	NUMBER(38)	Document ID dimension.	ASDIM_WC_ DOCUMENT_ 0.ID
USER_AGENT	NULL	NUMBER(38)	User agent.	ASDIM_WC_ USER_AG_0.ID
APPLICATION	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.20 ASFACT_WC_DOCLIB__1

Fact table that stores event information captured by the Analytics service when someone downloads a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	DATE	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
DOCUMENT	NULL	NUMBER(38)	Document ID dimension.	ASDIM_WC_ DOCUMENT_0.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_ REFERRE_0.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_ USER_AG_0.ID
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID.	
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.21 ASFACT_WC_DOCLIB__2

Fact table that stores event information captured by the Analytics service when someone creates document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	DATE	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.22 ASFACT_WC_DOCLIB__3

Fact table that stores event information captured by the Analytics service when someone edits a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.23 ASFACT_WC_DOCLIB__4

Fact table that stores event information captured by the Analytics service when someone tags a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
TAGS	NULL	NUMBER(38)	Tag added to the document.	
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.24 ASFACT_WC_DOCLIB__5

Fact table that stores event information captured by the Analytics service when someone "likes" a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	

I.2.25 ASFACT_WC_DOCLIB__6

Fact table that stores event information captured by the Analytics service when someone comments on a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.26 ASFACT_WC_DOCLIB__7

Fact table that stores event information captured by the Analytics service when someone deletes a document.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
WEBCENTER_ RESOURCE_	NULL	NUMBER(38)	WebCenter Portal resource ID.	

I.2.27 ASFACT_WC_EVENT_C_0

Fact table that stores event information captured by the Analytics service when someone creates an event in a space.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
EVENT_	NULL	NUMBER(38)	Event Identifier.	ASDIM_WC_ EVENTS_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.28 ASFACT_WC_EVENT_D_0

Fact table that stores event information captured by the Analytics service when someone edits an event in a space.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
EVENT_	NULL	NUMBER(38)	Event Identifier.	ASDIM_WC_ EVENTS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.29 ASFACT_WC_EVENT_E_0

Fact table that stores event information captured by the Analytics service when an event is deleted.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
EVENT_	NULL	NUMBER(38)	Event Identifier.	ASDIM_WC_ EVENTS_0.ID

I.2.30 ASFACT_WC_GROUPSP_0

Fact table that stores event information captured by the Analytics service when someone views a space.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
SESSION_ID_	NULL	NVARCHAR2 (254)	Session ID.	
RESPONSE_ TIME_	NULL	NUMBER(16)	Response time (ms).	
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_ REFERRE_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_ USER_AG_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.31 ASFACT_WC_GROUPSP_1

Fact table that stores event information captured by the Analytics service when someone creates a space in the Spaces application.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.32 ASFACT_WC_GROUPSP_2

Fact table that stores event information captured by the Analytics service when someone joins a space in the Spaces application.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.33 ASFACT_WC_GROUPSP_3

Fact table that stores event information captured by the Analytics service when someone deletes a space in the Spaces application.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.34 ASFACT_WC_LIST_CR_0

Fact table that stores event information captured by the Analytics service when someone creates a list.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
LIST_	NULL	NUMBER(38)	List ID.	ASDIM_WC_ LISTS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

I.2.35 ASFACT_WC_LIST_DE_0

Fact table that stores event information captured by the Analytics service when someone deletes a list.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
LIST_	NULL	NUMBER(38)	List ID.	ASDIM_WC_ LISTS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID

I.2.36 ASFACT_WC_LIST_ED_0

Fact table that stores event information captured by the Analytics service when someone edits a list.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.I D
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
LIST_	NULL	NUMBER(38)	List ID.	ASDIM_WC_LISTS_0.ID
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID

I.2.37 ASFACT_WC_LOGINS_0

Fact table that stores login event information captured by the Analytics service with related data for each event.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_EVENTFACTS.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
SESSION_ID_	NULL	NVARCHAR2(254)	Session ID	
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_USER_AG_0.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_REFERRE_0.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_CLIENT__0.ID

I.2.38 ASFACT_WC_PEOPLEC_0

Fact table that stores event information captured by the Analytics service when someone adds a people connection.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_EVENTFACTS.ID
TARGET_USER	NULL	NUMBER(38)	User ID	

I.2.39 ASFACT_WC_PEOPLEC_1

Fact table that stores event information captured by the Analytics service when someone posts on a wall.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
TARGET_ USER	NULL	NUMBER(38)	User ID.	

I.2.40 ASFACT_WC_PEOPLEC_2

Fact table that stores event information captured by the Analytics service when someone edits a profile.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID

I.2.41 ASFACT_WC_PEOPLEC_3

Fact table that stores event information captured by the Analytics service when someone edits status information.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.I D

I.2.42 ASFACT_WC_PORTLET_0

Fact table that stores information captured by the Analytics service when portlets are accessed by end users.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID
CACHED_	NULL	NUMBER(5)	Indicates whether the portlet was displayed from the cache.	
PORTLET_	NULL	NUMBER(38)	Portlet.	ASDIM_WC_ PORTLET_1.ID
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_ REFERRE_0.ID
PRODUCER_	NULL	NUMBER(38)	Portlet producer.	ASDIM_WC_ PRODUCE_0.ID
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_ PAGES_0.ID
SESSION_ID_	NULL	NVARCHAR2 (254)	Session ID.	
CLIENT_IP	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_ CLIENT__0.ID
PORTLET_ INSTANCE_	NULL	NUMBER(38)	Portlet instance.	
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_ GROUPSP_0.ID
RESPONSE_ TIME_	NULL	NUMBER(16)	Response time (ms).	
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_ APPLICA_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_ USER_AG_0.ID

I.2.43 ASFACT_WC_SEARCHE_0

Fact table that stores event information captured by the Analytics service when someone performs a search.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
OCURRED	NOT NULL	DATE	Date event occurred.	
TIMEID	NOT NULL	NUMBER(38)	Time dimension event occurred. This column is used for partitioning.	ASDIM_TIME.ID
USERID	NULL	NUMBER(38)	User ID dimension.	ASDIM_ USERS.ID
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASSYS_ EVENTFACTS.ID

Column Name	NOT NULL?	Data Type	Description	Foreign Key
PAGE_	NULL	NUMBER(38)	Page ID dimension.	ASDIM_WC_PAGES_0.ID
USER_AGENT_	NULL	NUMBER(38)	User agent.	ASDIM_WC_USER_AG_0.ID
CLIENT_IP_	NULL	NUMBER(38)	Client IP ID dimension.	ASDIM_WC_CLIENT__0.ID
APPLICATION_	NULL	NUMBER(38)	Application ID dimension.	ASDIM_WC_APPLICA_0.ID
SESSION_ID_	NULL	NVARCHAR2 (254)	Session ID.	
REFERRER_	NULL	NUMBER(38)	Referrer ID dimension.	ASDIM_WC_REFERRE_0.ID
SEARCHED_PHRASE	NULL	NUMBER(38)	Search phrase.	
GROUPSPACE_	NULL	NUMBER(38)	Space ID dimension.	ASDIM_WC_GROUPSP_0.ID

I.3 Analytics Dimension Table Descriptions

Analytics dimension tables are created when the Analytics Collector runs for the first time. Dimension tables store redundant descriptive data associated with analytics events. When a value is not found in these tables, a new record is added. Typical analytics dimension data includes: page name, browser name, document name, space name, and so on.

This section describes each of the tables listed in Table I–2, " Analytics Dimension Tables (Prefixed ASDIM_WC)".

I.3.1 ASDIM_WC_APPLICA_0

Dimension table that stores the names of WebCenter Portal applications accessing analytics data, for example `webenter` (Spaces application).

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
NAME_	NOT NULL	NVARCHAR2(254)	Application name.	

I.3.2 ASDIM_WC_CLIENT__0

Dimension table that stores client IP address details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
IP	NOT NULL	NVARCHAR2(15)	Client IP address	

I.3.3 ASDIM_WC_DISCUSS_0

Dimension table that stores discussion topic details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	
NAME_	NULL	NVARCHAR2(254)	Discussion topic name.	
DESCRIPTION_	NULL	NVARCHAR2(254)	Discussion topic description.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	

I.3.4 ASDIM_WC_DISCUSS_1

Dimension table that stores discussion forum details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	
DESCRIPTION_	NULL	NVARCHAR2(254)	Discussion forum description.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
NAME_	NULL	NVARCHAR2(254)	WebCenter Portal resource name, that is, the name of the discussion forum.	

I.3.5 ASDIM_WC_DISCUSS_2

Dimension table that stores discussion message details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date records last modified.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
NAME_	NULL	NVARCHAR2(254)	WebCenter Portal resource name, that is, the name of the discussion message.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
DESCRIPTION_	NULL	NVARCHAR2(254)	Discussion message description.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	

I.3.6 ASDIM_WC_DISCUSS_3

Dimension table that stores announcement details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
DESCRIPTION_	NULL	NVARCHAR2(254)	Announcement description.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
NAME_	NULL	NVARCHAR2(254)	WebCenter Portal resource name, that is, name of the announcement.	

I.3.7 ASDIM_WC_DOCUMENT_0

Dimension table that stores information captured by the Analytics service when documents are accessed by end users.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
CONTENTTYPE_	NULL	NVARCHAR2(254)	Document content type.	
CONTEXT	NULL	NVARCHAR2(512)	Document prefix (for identification purposes): ■ Document prefix - Folder URL for parent space ■ Blog prefix - Blog title name ■ Wiki prefix - Wiki title name	
NAME_	NULL	NVARCHAR2(254)	Document name.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
DESCRIPTION_	NULL	NVARCHAR2(254)	Document description.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	
PATH_	NULL	NVARCHAR2(512)	Document path.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
OBJECTTYPE_	NULL	NVARCHAR2(254)	Object type.	

I.3.8 ASDIM_WC_EVENTS_0

Dimension table that stores information about space events captured by the Analytics service when a user creates an event in his or her calendar.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
NAME_	NULL	NVARCHAR2(254)	WebCenter Portal resource name, that is, the name of the space event.	

I.3.9 ASDIM_WC_GROUPSP_0

Dimension table that stores information captured by the Analytics service when a space is accessed by an end user (Spaces application only).

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
ICONURL_	NULL	NVARCHAR2(254)	Icon URL	
DESCRIPTION_	NULL	NVARCHAR2(254)	Description.	
PERSONAL_	NULL	NUMBER(5)	Indicates whether the space is the Home space (0) or a group space (1).	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	
NAME_	NULL	NVARCHAR2(254)	Name of the space.	

I.3.10 ASDIM_WC_LISTS_0

Dimension table that stores list details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
NAME_	NULL	NVARCHAR2(254)	WebCenter Portal resource name, that is, the name of the list.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID.	

I.3.11 ASDIM_WC_PAGES_0

Dimension table that stores information captured by the Analytics service when pages are accessed by end users.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
PERSONAL_	NULL	NUMBER(5)	Indicates whether the page is a personal page (in the Home space) or belongs to a group space.	
RESOURCEID_	NOT NULL	NVARCHAR2(254)	WebCenter Portal resource ID. The full page path, for example, `/oracle/webcenter/page/sda254765_f950/Home.jspx`	
NAME_	NULL	NVARCHAR2(254)	Name of the page.	

I.3.12 ASDIM_WC_PORTLET_0

Dimension table that stores information captured by the Analytics service when portlets are accessed by end users.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record modified.	
DEFAULT_TITLE_	NULL	NVARCHAR2(254)	Default portlet title.	
RESOURCEID_	NOT NULL	NVARCHAR2(512)	WebCenter Portal resource ID.	

I.3.13 ASDIM_WC_PORTLET_1

Dimension table that stores portlet instance information captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
TITLE_	NULL	NVARCHAR2(254)	Portlet title.	
RESOURCEID_	NOT NULL	NVARCHAR2(512)	WebCenter Portal resource ID.	

I.3.14 ASDIM_WC_PRODUCE_0

Dimension table that stores portlet producer information captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record modified.	
RESOURCEID_	NOT NULL	NVARCHAR2(512)	WebCenter Portal resource ID.	
NAME_	NULL	NVARCHAR2(254)	Producer name.	

I.3.15 ASDIM_WC_REFERRE_0

Dimension table that stores information captured by the Analytics service related to the location (URL) to which the user was navigating when the analytics event was captured.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
URL	NOT NULL	NVARCHAR2(2000)	Referred URL	

I.3.16 ASDIM_WC_SEARCHE_0

Dimension table that stores search phrases captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
PHRASE	NOT NULL	NVARCHAR2(254)	Search phrase.	

I.3.17 ASDIM_WC_TAGS_0

Dimension table that stores tag information captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
VALUE	NOT NULL	NVARCHAR2(254)	Tag value.	

I.3.18 ASDIM_WC_USER_AG_0

Dimension table that stores information about whichbrowsers are used to access analytics data.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
OS	NULL	NVARCHAR2(100)	Operating system.	
AGENT	NOT NULL	NVARCHAR2(512)	User agent ID.	
BROWSER	NULL	NVARCHAR2(255)	Name of the browser.	
BROWSER_ VERSION	NULL	NVARCHAR2(30)	Browser version.	

I.3.19 ASDIM_TIME

Dimension table that stores times used to group analytics reports. An entry is created every hour. analytics reports use this table to group data within a specific time frame.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
PERIODSTART	NULL	DATE	Current period start.	
PERIODEND	NULL	DATE	Current period end.	
HOUROFDAY	NULL	NUMBER(5)	Hour corresponding to this time period.	
DAYOFMONTH	NOT NULL	NUMBER(5)	Current day of the month.	
DAYOFYEAR	NULL	NUMBER(5)	Current day of the year.	
WEEKOFYEARSU NDAY	NULL	NUMBER(5)	Week number of Sunday year.	
WEEKOFYEARMO NDAY	NULL	NUMBER(5)	Week number of Monday year.	
WEEKOFYEARSA TURDAY	NULL	NUMBER(5)	Week number of Saturday year.	
MONTHOFYEAR	NOT NULL	NUMBER(5)	Month corresponding to the period's year.	
YEAR	NOT NULL	NUMBER(5)	Current year.	

I.3.20 ASDIM_USERS

Dimension table that stores user details captured by the Analytics service when an analytics event is received.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
USERID	NOT NULL	NVARCHAR2(255)	User ID. For example: `weblogic`	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	

I.3.21 ASDIM_USERPROPERTIES

Dimension table that stores user properties, such as Title, Department, and so on.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
PROPERTYID	NOT NULL	NUMBER(38)	Property ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NULL	DATE	Date record last modified.	
NAME	NOT NULL	NVARCHAR2(255)	Property name.	

I.3.22 ASDIM_USERPROPERTYVALUES

Dimension table that stores values for user properties.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
USERID	NOT NULL	NUMBER(38)	User ID dimension.	ASDIM_USERS.ID
PROPERTYID	NOT NULL	NUMBER(38)	Property ID.	ASDIM_USERPROPERTIES.ID
VALUE	NULL	NVARCHAR2(255)	Property ID value.	
TYPE	NULL	NUMBER(38)	Property type.	

I.4 Analytics System Table Descriptions

System tables are created during installation to store application information such as system configuration parameters. Each table is described in this section.

I.4.1 ASSYS_CONFIG

System table that stores system configuration information. Each row contains value pairs—configuration type/value.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
NAME	NULL	NVARCHAR2(510)	Name of configuration parameter, for example, collector_port.	
CONFIGTYPEID	NOT NULL	NUMBER(38)	Configuration type ID.	
INTVAL	NULL	NUMBER(38)	Integer value associated with the configuration.	
STRVAL	NULL	NVARCHAR2(2000)	String value associated with the configuration.	

I.4.2 ASSYS_DIMENSIONPROPS

System table that stores dimension configuration information. Each row represents property values for a dimension.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
DIMENSIONID	NOT NULL	NUMBER(38)	Dimension ID.	ASSYS_EVENTDIMENSIONS.ID
DISPAYNAME	NOT NULL	NVARCHAR2(510)	Dimension property display name.	
COLUMNNAME	NOT NULL	NVARCHAR2(510)	Column name.	
COLUMNTYPE	NULL	NUMBER(38)	Column type.	
COLUMNLENGTH	NULL	NUMBER(38)	Column length.	
ISKEY	NULL	NUMBER(38)	Indicates whether this is a key dimension property.	

I.4.3 ASSYS_EVENTDIMENSIONS

System table that stores dimensions associated with each analytics event.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
DISPAYNAME	NOT NULL	NVARCHAR2(510)	Dimension event display name.	
TABLENAME	NOT NULL	NVARCHAR2(510)	Table name associated with the dimension.	
ISUNIQUE	NULL	NUMBER(38)	Deprecated.	
IDENTIFYINGUSER	NOT NULL	NUMBER(38)	User ID.	ASDIM_USERS.ID

I.4.4 ASSYS_EVENTFACTS

System table that stores facts associated with each analytics event.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
EVENTID	NULL	NUMBER(38)	Event ID dimension.	ASDIM_WC_EVENTS_0.ID
DISPAYNAME	NOT NULL	NVARCHAR2(510)	Event fact display name.	
COLUMNAME	NOT NULL	NVARCHAR2(510)	Event fact column name.	
COLUMNTYPEID	NULL	NUMBER(38)	Indicates the data type of the event fact: INTEGER (1), STRING (2), DATE (3), FLOAT (4), BOOLEAN (5) If the event fact is a dimension, COLUMNTYPEID is -1.	
COLUMNLENGTH	NULL	NUMBER(38)	User ID.	
EVENTDIMENSIONID	NULL	NUMBER(38)	Dimension ID.	ASSYS_EVENTDIMENSIONS.ID

I.4.5 ASSYS_EVENTS

System table that stores the different analytics events representing *one* user action that is captured. Both out-of-the-box analytics events and custom (user-defined) analytics events are stored. Each row represents the event captured with its associated data such as active status, and table name where data is stored.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
DISPAYNAME	NOT NULL	NVARCHAR2(510)	Event fact display name.	
DESCRIPTION	NULL	NVARCHAR2(510)	Event description.	
TABLENAME	NOT NULL	NVARCHAR2(510)	Table name.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ISCUSTOM	NOT NULL	NUMBER(38)	Indicates whether or not the event is a custom event. (0/1)	
ISACTIVE	NOT NULL	NUMBER(38)	Deprecated.	
ISPERSISTENT	NOT NULL	NUMBER(38)	Indicates whether or not to create a fact table for the event. (0/1)	
ISACTION	NOT NULL	NUMBER(38)	Not used.	

I.4.6 ASSYS_NAMESPACES

System table that stores the different namespaces in which analytics events are registered.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
PREFIX	NOT NULL	NVARCHAR2(510)	Namespace prefix.	
URI	NOT NULL	NVARCHAR2(510)	Universal resource identifier for the namespace.	

I.4.7 ASSYS_REPORTCOMPOSITION

System table that stores the composition for reports provided by the Analytics service.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
PARENTREPORTID	NOT NULL	NUMBER(38)	Reference to parent report ID.	ASSYS_REPORTS.ID
CHILDREPORTID	NOT NULL	NUMBER(38)	Reference to child report ID.	ASSYS_REPORTS.ID
SEQUENCE	NULL	NUMBER(10)	Sequence of report composition.	
NAME	NOT NULL	NVARCHAR2(510)	Name of report composition.	
DESCRIPTION	NULL	NVARCHAR2(510)	Description of report composition.	

I.4.8 ASSYS_REPORTGROUP

System table that stores the report groups included for the Analytics service. Report groups enable end users to display analytics data in a specific way.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
LASTMODIFIED	NOT NULL	DATE	Date record was last modified.	
NAME	NOT NULL	NVARCHAR2(510)	Report group name.	
DESCRIPTION	NULL	NVARCHAR2(510)	Report group description.	
SEQUENCE	NULL	NUMBER(10)	Sequence.	

I.4.9 ASSYS_REPORTITEMS

System table that stores item information in each report. Each row represents a specific item belonging to one report.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
REPORTID	NOT NULL	NUMBER(38)	Report ID.	ASSYS_REPORTS.ID
ITEMTYPE	NOT NULL	VARCHAR2(30)	Item type.	
ITEMDESCRIPTION	NULL	NVARCHAR2(510)	Item description.	
ITEMVALUE	NULL	NUMBER(5)	Item value.	
EVENTDIMENSIONID	NULL	NUMBER(38)	Dimension ID.	ASSYS_EVENTDIMENSIONS.ID
EVENTDIMENSIONNAME	NULL	NVARCHAR2(510)	Dimension name.	
EVENTFACTID	NULL	NUMBER(38)	Event fact ID.	ASSYS_EVENTFACTS.ID
EVENTFACTNAME	NULL	NVARCHAR2(510)	Event fact name.	
PROPERTYNAME	NULL	NVARCHAR2(510)	The name of the column in the dimension table to be selected in the query performed, for example, name or ID.	
AGGREGATEFUNCTION	NULL	VARCHAR2(60)	SQL function for grouping purposes (AVG, MIN, MAX, COUNT, COUNT DISTINCT).	
DISPLAYFLAG	NOT NULL	NUMBER(5)	Indicates whether or not to display the item in the report. (0/1)	
IDENTIFIERFLAG	NOT NULL	NUMBER(5)	Indicates whether or not to use this item as an identifier (SQL: whether or not to group by this column). (0/1)	

I.4.10 ASSYS_REPORTITEMVALUES

System table that stores the values associated to each item in a report. Each row represents a specific value (char, float or date) for one report item instance.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
REPORTITEMID	NOT NULL	NUMBER	Report item ID.	ASSYS_REPORTITEMS.ID
VALUETYPE	NOT NULL	VARCHAR2(30)	Value type: CVALUE, NVALUE or DVALUE.	
CVALUE	NULL	NVARCHAR2(510)	Char value.*	
NVALUE	NULL	NUMBER	Numeric value.*	
DVALUE	NULL	DATE	Date value.*	

*Only one of CVALUE, NVALUE or DVALUE has a value.

I.4.11 ASSYS_REPORTS

System table that stores the different out-of-the-box reports to be provided by the Analytics service. Each row represents a single report and the report's configuration.

Column Name	NOT NULL?	Data Type	Description	Foreign Key
ID	NOT NULL	NUMBER(38)	Internal table ID.	
CREATED	NOT NULL	DATE	Date record created.	
LASTMODIFIED	NOT NULL	DATE	Date record last modified.	
NAME	NOT NULL	NVARCHAR2(510)	Report item ID.	
DESCRIPTION	NULL	NVARCHAR2(255)	Report description.	
SEQUENCE	NULL	NUMBER(10)	Sequence.	
GROUPID	NULL	NUMBER (38)	Group ID for the report.	ASSYS_REPORTGROUP.ID
EVENTTYPEID	NULL	NUMBER(38)	Associated event ID.	ASSYS_EVENTS.ID
EVENTDISPLAYNAME	NULL	NVARCHAR2(510)	Denormalized. Display name of associated event.	
GROUPSPACEFILTERENABLED	NOT NULL	VARCHAR2(1)	Indicates whether users can filter the report 'by space'. If set to true (1), the report displays the space selector UI.	
DEFAULTCHARTSTYLE	NULL	NVARCHAR2(255)	Default chart style for the report.	
DEFAULTCHARTTYPE	NULL	NUMBER(2)	Default chart type.	
DEFAULTDISPLAYCOUNT	NULL	NUMBER(2)	Default display count.	
DEFAULTTIMEFRAME	NULL	NUMBER(2)	Default time frame.	

Column Name	NOT NULL?	Data Type	Description	Foreign Key
DEFAULTGROUPBYTIMEINTERVAL	NULL	NUMBER(2)	Default group time interval.	
DEFAULTGROUPBYUSERPROPERTY	NULL	NUMBER(38)	Default "group by user" property.	
SHOWMETRICSELECTOR	NOT NULL	NVARCHAR2(1)	Indicates whether or not to display selector UI for this metric. (Y/N)	
SHOWCHARTTYPEOPTIONS	NOT NULL	NVARCHAR2(1)	Indicates whether or not to display chart type options in the UI. (Y/N)	
SHOWDISPLAYOPTIONS	NOT NULL	NVARCHAR2(1)	Indicates whether or not the field to select display options will be shown in the UI. (Y/N) (The field is a drop-down list to select between: Top n, Bottom n, All records, or List of specific records.)	
SHOWDISPLAYTOPOPTION	NOT NULL	NVARCHAR2(1)	Indicates whether or not users will be able to select the "Top n" option. (Y/N) If N, the "Top n" option will not be available in the combo box.	
SHOWDISPLAYBOTTOMOPTION	NOT NULL	NVARCHAR2(1)	Indicates whether or not users will be able to select the "Bottom n" option. (Y/N) If N, the "Bottom n" option will not be available in the combo box.	
SHOWDISPLAYALLOPTION	NOT NULL	NVARCHAR2(1)	Indicates whether or not users will be able to select the "All records" option. (Y/N) If N, the "All records" option will not be available in the combo box, and users will only be able to select from Top n, Bottom n or a list of specific records.	
SHOWOBJECTPICKER	NOT NULL	NVARCHAR2(1)	Indicates whether or not users will be able to select a list of specific records option. (Y/N) If N, the "Select ..." option will not be available in the combo box. If Y the "Select ..." option will be available and a picker dialog will be displayed to select the list of objects to be included in the report.	
SHOWTIMEFRAMEFILTER	NOT NULL	NVARCHAR2(1)	Indicates whether or not to display time frame filter. (Y/N)	
SHOWUSERPROPERTYFILTER	NOT NULL	NVARCHAR2(1)	Indicates whether or not to display user property filter (Y/N).	
SHOWGROUPBYOPTIONS	NOT NULL	NVARCHAR2(1)	Indicates whether or not to display "group by" options. (Y/N)	
COMPOSEDREPORT	NOT NULL	NVARCHAR2(1)	Indicates whether this is a composed report.	
PORTLETTYPEID	NULL	NUMBER(38)	Portlet type ID.	

I.5 Analytics User Properties

The Analytics service uses several tables to store information about users who are triggering analytics events in WebCenter Portal applications. This section contains the following topics:

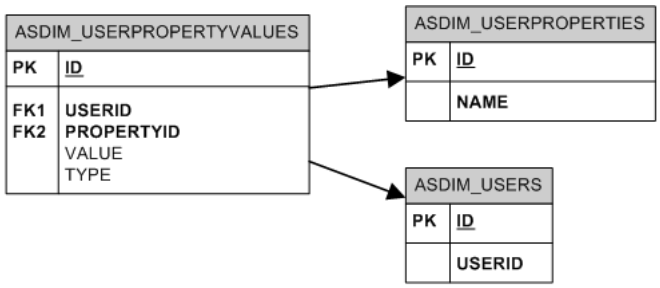
- Section I.5.1, "User Property Tables - Overview"
- Section I.5.2, "User Property Dimension Tables"
- Section I.5.3, "Out-of-the-Box User Properties for WebCenter Portal"

For sample queries related to user properties, see Section 47.3.3, "Sample Queries for User Metrics" in Chapter 47, "Integrating the Analytics Service."

I.5.1 User Property Tables - Overview

Figure I–5 shows the database model used by the Analytics service to store information about users. The user dimension table has a single property column (USERID). In addition, there are two other tables (ASDIM_USERPROPERTIES and ASDIM_USERPROPERTYVALUES) that store user properties as key-value data. This model enables additional user properties to be added without changing the user dimension definition.

Figure I–5 User Property Dimensions for Analytics



The ASDIM_USERPROPERTIES table stores the names of user properties received by the Analytics service. User properties (attributes of the user including name, position, email, address, etc) are captured by the Analytics service as an event when a user logs in to Spaces, not when the properties are modified. If a user changes a property, the value of the property will be updated in analytics the next time the user logs in.

The ASDIM_PROPERTYVALUES table stores one row per property, per user, where each row corresponds to a value for a property associated with a specific user.

I.5.2 User Property Dimension Tables

User property dimension tables are described in the following topics:

- Section I.5.2.1, "ASDIM_USERPROPERTIES"
- Section I.5.2.2, "ASDIM_USERPROPERTYVALUES"

I.5.2.1 ASDIM_USERPROPERTIES

Dimension table that stores the names of user properties received by the Analytics service.

Column Name	Data Type	Description
ID	NUMBER(38,0)	Table primary key. Value obtained from the sequence.
NAME	VARCHAR(255)	Name of the user property. Names are converted to uppercase and whitespace characters are removed.

I.5.2.2 ASDIM_USERPROPERTYVALUES

Dimension table that stores one row per property per user, where each row corresponds to a value for a property associated with a specific user.

Column Name	Data Type	Description
ID	NUMBER(38,0)	Table primary key. Value obtained from the sequence.
USERID	NUMBER(38,0)	Foreign key to the ASDIM_USERS table.
PROPERTYID	NUMBER(38,0)	Foreign key to the ASDIM_USERPROPERTIES table.
VALUE	VARCHAR(255)	The value of a particular property (identified by the PROPERTYID column) for a particular user (identified by the USERID column). For example, if the property is "CITY" and the user "weblofig", this column will be storing the city name where that user works (for example "San Francisco").
TYPE	NUMBER(38,0)	User property value type. (Property values are always stored as strings.)

I.5.3 Out-of-the-Box User Properties for WebCenter Portal

WebCenter Portal applications send a specific set of user properties to the Analytics service, as shown in Table I–4.

Note: WebCenter Portal applications can only send properties that are set by the user or available in the user directory. If a property value is blank, no row is created for it in the ASDIM_USERPROPERTYVALUES table for that user.

Table I–4 Out-of-the-Box User Properties for WebCenter Portal

Property Name	Property Label (displayed in reports)	Example Value
IMUSER	IM User	dfrabott
DISPLAYNAME	Display	Diego Frabotta
PHONE	Phone	+5411-2222-1234
TITLE	Title	Software Engineer
DEPARTMENT	Department	Analytics
MANAGER	Manager	uid=dsabaris,cn=users,dc=us,dc=myco,dc=com
COMPANY	Company	MyCo
STREET*	Street	Parque Austral, Edificio Insignia M3
ZIPCODE	ZIP Code	12345

Table I–4 (Cont.) Out-of-the-Box User Properties for WebCenter Portal

Property Name	Property Label (displayed in reports)	Example Value
STATEORPROVINCE	State or Province	Buenos Aires
COUNTRY	Country	Argentina
EMPLOYEEID	Employee Id	55555
CITY	City	Pilar

* STREET stores the user's home address street.

I.6 Analytics Event and Dimension Data Types

Table I–5 shows how analytics event and dimension data types map to data types in Oracle, Microsoft SQL Server, and IBM DB2 databases.

Table I–5 Analytics Event and Dimension Data Types

Analytics Event and Dimension Type	Oracle	Microsoft SQL Server	IBM DB2
INTEGER	NUMBER	BIGINT	INTEGER
STRING	NVARCHAR	NVARCHAR	VARGRAPHIC
DATE	DATE	DATETIME	TIMESTAMP
FLOAT	FLOAT	FLOAT	DECFLOAT
BOOLEAN	NUMBER(5)	TINYINT	SMALLINT

Glossary

About mode

A **portlet mode** that typically displays information such as copyright, version, and author of the portlet.

Activity Stream

In the **People Connections service**, a feature for viewing the application activities tracked for you and other users.

Activity Graph

A WebCenter Portal service that provides suggestions of people, items, and spaces that users may be interested in interacting with.

The engine used by the Activity Graph service to provide a central repository for actions that are collected by enterprise applications. The data stored in the activity graph is analyzed to calculate ranks for nodes, predict new actions, and make recommendations.

Activity Rank

Determines the relevance of a search result.

administrator

In the Spaces application there are two types of administrator:

- Fusion Middleware administrator: Also referred to as systems administrator. A user with complete administrative capabilities. This administrator can perform the complete range of security-sensitive administrative duties, and all installation, configuration, and audit tasks.
- Spaces administrator: A user responsible for customizing Spaces out of the box, managing and granting application roles, and maintaining the application when it is in use.

Ajax

A combination of asynchronous JavaScript, dynamic HTML (DHTML), XML, and XmlHttpRequest communication channel that enables requests to be made to the server without fully re-rendering the page. Ajax enables rich client-like applications to use standard internet technologies.

Analytics service

A WebCenter Portal service offers real-time usage and activity reporting for your portal. In the Spaces application, users can track and analyze Spaces traffic and usage.

Announcements service

A WebCenter Portal service that offers a quick, convenient way to create and widely distribute messages instantly or at a specific time.

API

Application Programming Interface. A set of exposed data structures and functions that an application can use to invoke services on an application object, such as a **portlet**.

application customization

Performed by an administrator, all users see the change. These are static changes to an application that affect a site or sites that do not involve changes to the application's code or schema.

See also **user customization** and **personalization**

Application Development Framework

See **Oracle ADF**.

Application Programming Interface

See **API**.

application role

Roles that are specific to a particular application and are stored in an application-specific stripe of the policy store.

application skin

Specifies the application background color, screen fonts, and, with some skins, the shapes and images used for application buttons and icons. In the Spaces application, the administrator chooses the default application skin and Spaces users may change the application skin on the General tab of the Preferences dialog.

application templates

WebCenter Portal provides templates for creating two kinds of applications: Framework applications and Portlet Producer applications. Templates ensure that the right technology scopes are set, tag libraries added, and required Java classes are added to the class path. Once you do this, relevant components are included to the Component Palette and relevant context menus become available in JDeveloper.

See also **Framework application template** and **Portlet Producer Application template**.

authenticated user

A user who is logged into a WebCenter Portal application. Credentials of this user are verified against the identity store. By default, an authenticated user can access public information. To access secured information, such as pages and **portlet**s, this user must be authorized through the policy and credential store.

Contrast with **public user**s, who are not logged in, and can access public content only.

authentication

Identification of a user through an identity management system. You can require ADF authentication to enforce credentials for users to access the Framework application only (all ADF resources in the application remain accessible), or authentication *and*

authorization to enforce credentials for users to access the Framework application and any ADF resources that have been secured in the application.

authorization

The policies that define the access rights of an individual or group to a secured resource. This resource may be a page or component within a page.

authorized user

An individual who has access to a secured resource. For non-public resources, this individual is also an **authenticated user**.

AviTrust Portal Demonstration for WebCenter Portal

An enterprise banking application built using Oracle Fusion Middleware, including Oracle WebCenter Portal: Framework and WebCenter Portal: Spaces, used to provide examples of WebCenter Portal functionality.

blog page

A page that provides a personal record of an individual user's experience and opinions. There are two kinds of blog: personal blogs are written by an individual; group blogs are written by several users.

Box layout component

A layout component available through **Composer**. A container that enables the placement of content on a page created in the Spaces application. In Composer, a Box is rendered as a rectangle comprised of dashed lines. In a Framework application, this is the runtime equivalent of a Panel Customizable component.

BPEL

Business Process Execution Language. An XML-based markup language for composing a set of discrete web services into an end-to-end process flow.

business role page

A page, created by the administrators in the Spaces application, specifically provided for a given role in an organization. Business role pages provide a targeted environment for users of a particular role by delivering information that is timely and relevant to individual roles without the noise of irrelevant information from other lines of business. Business role pages appear in the Home space of users classified under the specified role.

caching

The act of storing frequently accessed information, typically web pages, in a location where it can be accessed quickly to avoid frequent content generation.

See also **expiry-based caching** and **validation-based caching**.

calendar overlay

The ability to display multiple calendars in a single Events task flow.

Change Mode Button component

In the **Composer** tag library that enables users to change from page View mode to page Edit mode.

Change Mode Link component

A component provided in the **Composer** tag library that enables users to change from page View mode to page Edit mode.

check out/check in

A mechanism that enables a user to lock information, by checking it out, so that other users cannot modify that same piece of information. This prevents users from overwriting each other's changes. After making modifications, the user releases it by checking it back in, making it available again for other users to modify.

Child Components

The components contained within a parent component. For example, the task flows contained within a Box layout component are the child components of the Box.

See also **Box layout component** and **parent component**.

chrome

Visual elements surrounding a portlet or task flow that provide an access point for actions, such as those on the Actions menu and those embedded in the chrome itself, such as the minimize icon or resize handles.

CMIS

Content Management Interoperability Services (CMIS) standard defines a domain model and Web services and Restful AtomPub bindings that can be used by applications to work with one or more Content Management repositories or systems.

component

An individual piece of an application, for example, a task flow, portlet, page, or layout element such as a box or image.

Component Catalog

A dialog, accessed from **Composer**, that provides access to all the content you can add to a WebCenter Portal application page.

component developer

The developer who builds components (such as portlets, **JavaServer Faces** components, and web services).

Component Properties

A dialog, accessed from **Composer**, that provides access to a component's parameters, display options, child components, style settings, and associated events.

Composer

A seamlessly integrated environment for populating, revising, and configuring portal pages. Composer enables users to easily build or revise page layout and content. It also provides the means of adding different components, such as task flows, portlets, content, and other objects, onto a page and then linking those components for a more relevant or personalized view of the information.

Connections

In the **People Connections service**, a feature for establishing a social network with other application users.

container

An application program or subsystem in which the program building block, known as a **component**, is run.

container runtime option

A JSR 286 feature that provides a way to customize the behavior of the portlet container and therefore customize the runtime environment.

content integration services

Services provided by **Oracle WebCenter Portal** to enable developers to display content from a **content repository**, such as by creating **data control**s.

Content Presenter

A feature of the **Documents service** that enables end users to select and search content items and then display those items using available display templates. Oracle WebCenter Portal provides out-of-the-box templates for displaying single and multiple content items on your pages. You can also define custom templates for the content that you want to display in your **Framework application**, or for selection by end users at runtime.

content repository

A specialized storage and management mechanism that provides such features as author-based versioning, full text searching, and content categorization and attribution. A content repository is optimized for storing unstructured information, which differentiates it from a data repository.

content repository data control

A **data control** sourced though a content repository. In a **Framework application**, you can create content repository data controls for the following content repositories: **Oracle Portal**, **Oracle WebCenter Content**, third-party repositories that support the Java Content Repository (JCR) standard, and your local file system.

credential provisioning page

A **JSF** (`*.jspx`) page used for authenticating to an **external application**. At runtime, the Credential Provisioning page displays login data fields consisting of the data fields specified through external application registration. Login information is passed to the producer, which in turn passes the login values to the external application. The application provides the producer with the requested portlets.

After authentication, the user's login credentials are preserved in a **credential store**, which subsequently supplies that information at future sessions. Unless his information changes, the user supplies his credentials only one time.

credential store

Provides storage for login credentials for its associated domain. It also preserves the login credentials that a user provides for authentication to an **external application**. Credential store is usually combined with the policy store as a single logical store.

Although the credentials stored in the credential store are used during subsequent logins for authentication, the main function of this store is to provide authorization for those accounts.

CSS

Cascading Style Sheet. A simple mechanism for ensuring a consistent look and feel or adding style, such as fonts, colors, and spacing, to web documents.

custom action

Icons or menu items rendered on the header or the Actions menu of a Show Detail Frame component surrounding a task flow. Custom actions can represent actions that were defined in the task flow when it was created. For example, at design time a developer can build a task flow with custom personalization settings. At runtime, users can access these settings through icons or Actions menu items provided in the task flow's surrounding chrome (or Show Detail Frame).

custom attribute

In the Spaces application, custom attributes specify information in addition to that provided by the built-in attributes. Custom attributes can be used to determine the content of the components in a space based on the parameter passed in. For example, a component can display data for a specific customer by passing in the customer ID. A custom attribute is simply a name value pair, for example customerId=400, orderId=11, userName=Smith, and so on. Custom attributes are stored within the space template.

custom page

Any page created by a user rather than one provided out of the box.

custom display template

A Content Presenter display template is a JSFF file (JSF page fragment) that defines how Content Presenter renders content items on a Framework application page. WebCenter Portal provides several out-of-the-box display templates to get you started, or, you can create your own templates.

custom resource catalog

A resource catalog that has been customized to control the components that are visible to specific users.

Contrast with **default resource catalog**.

custom role

A user role created by an administrator or a space moderator to meet a specific Home space or space requirement.

Customize mode

A **portlet mode** that enables users to set the default values for portlet preferences for all users.

customizable component

A WebCenter Portal component that can be added to a page at runtime to enable end users to perform personalizations such as move, minimize, restore, or remove on content within those components. Customizable components are the **Panel Customizable component** and the **Show Detail Frame component**.

data control

A mechanism that provides an abstraction of the business service's data model. The ADF data controls provide a consistent mechanism for clients and web application controllers to access data objects, collections, methods, and operations.

default language (application-level)

A display language specified by the Spaces administrator that is used when users log in to the Spaces application. The Spaces administrator sets the application-level default

language on the **General** tab of the **Administration** page. Individual users can set their own user-level default language on the General tab of the Preferences dialog.

default language (user preference)

A user-specified display language that is rendered when the user logs in to the Spaces application. This language selection lasts until the user specifies a different default language. It can be overridden by a session language, but returns as the default when the session cookie is purged or expires. This value is set on the **General** tab of the Preferences dialog.

default resource catalog

The resource catalog that is provided by default for an application. It contains all of the Oracle ADF components and portlets available to the application.

Contrast with **custom resource catalog**.

Default Server

See **Integrated WLS**.

delegated administration

Provides a mechanism for securing portal resources based on user roles. You apply delegated administration to a page hierarchy, and the specific security assignments are automatically propagated down through the hierarchy through pages and sub pages.

deployment profile

A file used in application deployment that specifies the following types of information:

- The source files, deployment descriptors, and other auxiliary files that are packages
- The type and name of the archive file to be created
- Dependency information
- Platform-specific instructions
- Other information

Oracle WebCenter Portal: Services provides a special deployment profile, the **Framework application** WAR deployment profile, that includes an option to export project metadata.

Design view (JDeveloper)

A view, in **Oracle JDeveloper**, that provides a WYSIWYG representation of a file.

See also **Source view (JDeveloper)**.

Design view (Spaces)

A view, in **Composer**, that provides a WYSIWYG representation of a page and its components.

See also **Source view (Spaces)**.

discoverable space

A space that can be found by anyone logged into the Spaces application, for example through a search. Any **Public** or **Private** space is discoverable. Discoverable spaces are listed on the **Spaces** page when **All Spaces** is selected from the **Show** list. Users

wishing to join the space can request membership through self-service (if enabled) or by contacting the space moderator.

Discussions service

A WebCenter Portal service that provides a means of creating and participating in discussion forums.

display language

Controls the language in which application user interface elements, such as buttons, field labels, and screen text, are rendered in the browser. The order of precedence for Framework application display language settings from weakest to strongest is: browser setting, application setting, user preference setting, session setting, space setting.

Documents page

A predefined page provided in every Spaces group and Home space that includes the Document Manager task flow for managing content.

Documents service

A WebCenter Portal service that provides several task flows that offer a variety of formats to display folders and files on a page. You can choose the task flows appropriate for your application to provide features for accessing, adding, and managing folders and files; configuring and viewing file and folder properties; and searching file and folder content in the connected content repositories.

domain

Any tree or subtree within the Domain Name System (DNS) namespace. Domain most commonly refers to a group of computers whose host names share a common suffix: the domain name.

dynamically-generated page

A page that displays as the result of a user action, such as a search or a click on a tag. As the name suggests, dynamically-generated pages are not stored, but rather are created as and when needed.

EAR

Enterprise Archive file. A **Java EE** archive file that is used in deploying applications on a **Java EE** application server. **Framework application**s are deployed using both a generic EAR file, which contains the application and the respective runtime customization, and a targeted EAR file, which contains only the application for deployment to the application server. EAR files simplify application deployment by reducing the possibility of errors when moving an application from development to test, and test to production.

See also **WAR**.

ECMA-262 specification

A standardization of scripting programming languages, such as **ECMAScript** and JavaScript.

ECMAScript

A scripting programming language, standardized by Ecma International according to the **ECMA-262 specification**. Frequently referred to as JavaScript or JScript, which are both extensions of the ECMA-262 specification.

Edit Defaults mode

(**JSR 286** portlets only.) A **portlet mode** that enables personalization of a JSR 286 portlet. Edit Defaults mode is a display mode for the JSR 286 portlet's properties. In a **Framework application**, the Edit Defaults mode displays on the portlet's Actions menu as the Customize command.

See also **Edit mode**.

Edit mode

A **portlet mode** that enables personalization of the portlet for each user, for each instance.

See also **Edit Defaults mode**.

edit mode

A view mode that enables users to modify the content, style, and layout of a page. Access edit mode by choosing Edit Page from the Page Actions menu.

EL

Expression Language. Provides a shorthand way of working with web application data by providing operators for retrieving and manipulating application data residing in a **Java EE** web container. In a **Framework application**, EL expressions are encapsulated in the characters "#{" and "}" and typically come in the form #{object.data} where *object* represents any Java object or **Oracle ADF** component placed in the **Java EE** web container's page, request, session, or application's scope.

Enterprise Archive file

See **EAR**.

enterprise mashup

An application that enables users to bring all sorts of content and services together in a single place.

Events service

A WebCenter Portal service that provides calendars for scheduling meetings, appointments, and so on. In the Spaces application, it provides calendars to record events relevant to the specific space. You can also integrate the Events service with Microsoft Exchange Server to enable individual users to access their personal calendars in their Home space. Personal calendars are also available in Framework application.

expiry-based caching

A **caching** method that uses a retention period to specify how long the item is valid in the cache before a refresh is required. When there is a request for the item beyond the retention period, it is refreshed in the cache.

See also **validation-based caching**.

Expression Language

See **EL**.

external application

Applications that do not delegate authentication to the single sign-on server. Instead, they display HTML login forms that ask for application user names and passwords. At

the first login, users can choose to have the single sign-on server retrieve these credentials for them. Thereafter, they are logged in to these applications transparently.

farm

A collection of components managed by Fusion Middleware Control. A farm can contain a Managed Server domain and other Oracle Fusion Middleware system components that are installed, configured, and running on the domain.

favorites

A personal list of links to favorite pages in the Spaces application and external web sites.

Federated Portal Adapter

See **FPA**.

Feedback

In the **People Connections service**, a feature for posting informal appraisals for and receiving informal appraisals from other application users.

FPA

Federated Portal Adapter. A component of **Oracle Portal** that enables Oracle Portal instances to share their database portlets through the web portlet interface. Using the FPA, Oracle Portal database portlets, including PL/SQL portlets, Portlet Builder portlets, and page portlets can be made available for use in Framework applications.

Framework application

A Framework application is built on top of the ADF using the **WebCenter Portal's Extension for Oracle JDeveloper**. This application combines web content, portlets, content integration, and collaborative services for the end user. Developers and administrators can create a Framework application based on their roles and skill levels in the organization.

A portal also includes page hierarchies, navigation models, and delegated administration.

Framework application administrator

The administrator responsible for maintaining the **Framework application**. For example, in Spaces, the administrator performs tasks such as implementing the branding for the Framework application, making new content available, modifying pages, and granting and revoking privileges.

Contrast with systems administrator who has administrative rights for entire Fusion Middleware functions. The Fusion Middleware administrator is also responsible for deploying, setting up, and configuring the Framework application, and performing on-going administrative tasks for the Framework application and other WebCenter Portal components through Fusion Middleware Control.

Framework application developer

The developer who plans, builds, and maintains a **Framework application** using the Oracle Application Development Framework, **Oracle JDeveloper**, and **Oracle WebCenter Portal**.

Framework application end user

The runtime user of the **Framework application**, who accesses pages, portlets, and content, and personalizes portlets (assuming the appropriate privileges).

Framework application life cycle

The process of creating and testing a Framework application in a design time environment, deploying it to a production system, and then performing routine maintenance, such as monitoring performance and migrating customization data. The life cycle of an application also includes performing further enhancements, restaging, and then redeploying the application to the production system.

Framework application template

A JDeveloper template which includes WebCenter Portal: Framework features like site navigation, page hierarchies, delegated administration, and page templates.

See also **application templates** and **Portlet Producer Application template**.

Full Screen Mode (Spaces)

A view mode that opens the space to occupy the entire screen, thus maximizing the display space. The Sidebar is not displayed in Full Screen Mode.

Full Screen mode (Portlets)

(**PDK-Java** portlets only.) A **portlet mode** that provides more content than can be shown in the portlet when it is sharing a page with other portlets.

Fusion Middleware Control

A browser-based management application that is deployed when you install Oracle WebCenter Portal. From Fusion Middleware Control, you can monitor and administer a **farm** (such as Oracle WebCenter Portal).

Fusion Order Demo (FOD)

See **AviTrust Portal Demonstration for WebCenter Portal**.

HA

High Availability. A collection of solutions to ensure that your applications meet the required availability to achieve your business goals, eliminating single points of failure with no or minimal outage in service.

Help mode

A **portlet mode** that displays usage information about the functionality of the portlet.

High Availability

See **HA**.

Home space

A work area within Spaces that provides individual users with a private space for storing personal content, keeping notes, viewing and responding to assignments, maintaining a list of online buddies, and performing many other tasks relevant to their unique working day. Users can also extend this environment by creating additional personal pages and custom content.

HTML Markup layout component

A layout component available through **Composer**. A simple HTML component that renders raw HTML and JavaScript mark-up inline on the page.

Hyperlink layout component

A layout component available through **Composer**. A link to an internal or external web page. For designers of Framework applications, this is the runtime equivalent of a Go Link component.

IDE

Integrated Development Environment. A visual application development tool containing editors, debuggers, screen painters, object browsers, and the like. **Oracle JDeveloper** is an example of an IDE.

Identity Propagation

For a Framework application and associated content repositories, selecting this option allows propagation of current user's identity across the application and processes. The propagated identity is verified on the receiver's side, and then it is used to make decisions such as assigning role based access control.

Image layout component

A layout component available through **Composer**. An illustration that can include a hyper link. For designers of Framework applications, this is the runtime equivalent of an Image Link component.

IMP service

See **Instant Messaging and Presence service**.

initialization parameters

The parameters initialized upon the start-up of a standard JSR 286 portlet. Initialization parameters provide an alternative to JNDI (Java Naming and Directory Interface) variables. Use initialization parameters instead of JNDI to configure the behavior of all of the different components of the portlet—for example, servlets and other portlets—in a compatible way. In **Oracle WebCenter Portal**, initialization parameters are entered into the `portlet.xml` file.

Instant Messaging and Presence service

A WebCenter Portal service that enables users to observe the presence status of other authenticated users and provides instant access to interaction options, such as instant messages and emails.

Integrated Development Environment

See **IDE**.

Integrated WLS

Integrated WebLogic Server. A WLS instance used as a platform for pretesting Framework application deployments on a local computer. Integrated WLS also contains preconfigured portlet producers and several useful prebuilt portlets.

Iterative development

Iterative development lets you make changes to your Framework application while it is running on the Integrated WebLogic Server and immediately see the effect of those changes simply by refreshing the current page in your browser. The iterative development feature works by disabling certain optimization features. Iterative development allows developers to work more quickly and efficiently when building a Framework application.

JAAS

Java Authentication and Authorization Service (JAAS) is a Java package that enables applications to authenticate and enforce access controls upon users. JAAS is designed to complement Java 2 security and implements a Java version of the standard Pluggable Authentication Module (PAM) framework. This enables an application to remain independent from the authentication service, and supports the use of custom authentication modules.

JAAS extends the access control architecture of the Java 2 Security Model to support subject-based authorization. It also supports declarative security settings in deployment descriptors instead of being limited to code-based security settings.

Java Authentication and Authorization Service

See **JAAS**.

Java Content Repository

See **JCR 1.0**.

Java EE

Also known as Java EE 5. Java Enterprise Edition 5 Platform. A platform that enables application developers to develop, deploy, and manage multitier, server-centric, enterprise-level applications. The Java EE platform offers a multitiered distributed application model, integrated XML-based data interchange, a unified security model, and flexible transaction control. You can build your own Java EE portlets and expose them through web producers.

Java Enterprise Edition 5 Platform

See **Java EE**.

Java Portlet Specification

Standardizes how components for portal servers are to be developed. This specification defines a common portlet **API** and infrastructure that provides facilities for personalization, presentation, and security. Portlets using this **API** and adhering to the specification are product-agnostic, and can be deployed to any portal product that conforms to the specification. See also **JSR 286**.

Java Specification Request

See **JSR 286**.

JavaServer Faces

See **JSF**.

JavaServer Page

See **JSP**.

JCR 1.0

Java Content Repository 1.0. Also known as JSR 170. It proposes a standard access and interaction **API** for content repositories, much like JDBC does for databases.

JDeveloper

See **Oracle JDeveloper**.

JSF

JavaServer Faces. A standard Java framework for building web applications. It simplifies development by providing a component-centric approach to developing Java web user interfaces. JSF offers rich and robust **API**s that provide programming flexibility and ensures that applications are well designed with greater maintainability by integrating the Model-View-Controller (**MVC**) design pattern into its architecture. As JSF is a Java standard developed through Java Community Process, development tools like **Oracle JDeveloper** are fully empowered to provide easy to use, visual, and productive development environments for JSF.

JSF JSP

JavaServer Faces JavaServer Page. JSF JSPs differ from plain JSPs through their support of **Oracle ADF Faces** components for the user interface and JSF technology for page navigation. JSF JSP pages leverage the advantages of the Oracle **Application Development Framework** (Oracle ADF) by using the ADF Model binding capabilities for the components in the pages.

JSP

JavaServer Page. An extension to servlet functionality that provides a simple programmatic interface to web pages. JSPs are HTML pages with special tags and embedded Java code that is executed on the web or application server. JSPs provide dynamic functionality to HTML pages. They are actually compiled into servlets when first requested and run in the servlet container.

See also **JSP tags**.

JSP tags

Tags that can be embedded in **JSP**s to enclose Java code. These tags use the `<jsp:` syntax and enclose action elements in the JSP with `begin` and `end` tags similar to XML elements.

JSR 286

Java Specification Request (JSR) 286. Defines a set of **API**s for building standards-based portlets using Java. Portlets built to this specification can be rendered to a portal locally or deployed to a WSRP container for rendering portlets remotely. For more information, see `http://jcp.org/en/jsr/detail?id=286`.

JSR 170

See **JCR 1.0**

JSR 329

See **Oracle JSF Portlet Bridge**.

keystore

A file that provides information about available public and private keys that are used for authentication and data integrity. User certificates and the trust points needed to validate the certificates of peers are also stored securely in the keystore.

layout box

A container that enables placement of content on a page created in the Spaces application.

layout component

An object for enhancing the usefulness and appearance of a given page. Layout components include layout boxes, a rich text editor, images, hyperlinks, and so on.

Layout Customizable component

A component provided in the **Composer** tag library that enables users to select from a set of predefined layouts (for example, two column, three column, two row, and so on) and apply it to the page. Users can apply these layouts to a particular area of the page or to the entire page.

LDAP

Lightweight Directory Access Protocol. A standard, extensible directory access protocol. It is a common language that LDAP clients and servers use to communicate. The framework of design conventions supporting industry-standard directory products, such as the Oracle Internet Directory.

life cycle

See **Framework application life cycle**.

Lightweight Directory Access Protocol (LDAP)

See **LDAP**.

Links service

A WebCenter Portal service that provides a means of creating a bidirectional association between two objects, thus setting up easy access between those objects.

Lists service

A WebCenter Portal service that provides a means of creating lists and exposing them for placement on application pages at runtime. At design time, you can make the Lists task flow available in your runtime Resource Catalog. At runtime, users can add the task flow from the Catalog to a page and use the task flow to create lists.

Lists page

A predefined page that displays the space's current lists.

Look and feel

A look and feel file determines the appearance of your application, from the placement and behavior of elements on a portal page to the colors used in the portlet title bars. The look and feel is determined by skins, navigations, page templates, layouts, content display templates, and other similar components.

Mail service

A WebCenter Portal service for exposing familiar email functionality in applications.

Managed Server

In a production environment, a Managed Server hosts applications and the resources needed by those applications. A domain, which is a logically related group of Oracle WebLogic Server resources, can have any number of Managed Servers. An Administration Server manages these servers.

mashup

A web application that enables end users to pull information from different sources to create a personalized application that exactly meets their individual requirements.

See also **enterprise mashup**.

MBean Browser

In Fusion Middleware Control, MBean browsers enable the administrator to perform specific monitoring and configuration tasks and browse MBeans for an Oracle WebLogic Server or a selected application.

MDS

Oracle Metadata Services. A core technology of the **Application Development Framework**. MDS provides a unified architecture for defining and using metadata in an extensible and customizable manner.

MDS repository

An application server and Oracle relational database that keep metadata in these areas: a file-based repository, dictionary tables accessed by build-in functions, and a metadata registry. One of the primary uses of MDS is to store customizations and persisted personalization for Oracle applications.

Message Board

In the **People Connections service**, a feature for posting messages to and receiving messages from other application users.

metadata

Information about a content item, such as title, author, or security group. Metadata is used to describe, find, and group content items. Also referred to as content information.

Model-View-Controller

See **MVC**.

moderator

A Spaces user who is responsible for managing a particular space. A space moderator can add and remove members, invite new members, enable self registration, provide and update space metadata, and manage the services available to the space.

Movable Box layout component

A layout component available through **Composer**. A container that enables the placement of content on a page created in the Spaces application. Movable Boxes, along with their content, can be moved around on the page. For designers of Framework applications, this is the runtime equivalent of Show Detail Frame component.

MVC

Model-View-Controller. A classic design pattern often used by applications that need the ability to maintain multiple views of the same data. The MVC pattern hinges on a clean separation of objects into one of three categories: models for maintaining data, views for displaying all or a portion of the data, and controllers for handling events that affect the model or views. Because of this separation, multiple views and controllers can interface with the same model. Even new types of views and controllers that never existed before, such as portlets, can interface with a model without forcing a change in the model design.

My Spaces page

A predefined page that displays a list of all the spaces available to the currently logged in user. The user can select from the **Show** menu to display **All Spaces**, only spaces of which the user is a member (**Joined by Me**), or only spaces of which the user is the moderator (**Moderated by Me**).

navigation

WebCenter Portal provides three navigation components to create portal navigation. These components are: Breadcrumb navigation, menu navigation, and tree navigation.

navigation model

Navigation models provide data to the navigation user interface and enable navigation to resources in your application, such as pages, page hierarchies, task flows, external sites, portlets, and other entities. You can configure navigation models at both design time and runtime.

Notes service

A WebCenter Portal service that provides useful features for writing personal notes and reminders. This service is available only in Spaces, and not in Framework applications.

Notifications service

A WebCenter Portal service that provides an automated means of triggering notices across different messaging channels, such as phone, mail, Worklist, and so on. Messages are triggered when the spaces and application objects to which you have subscribed change.

Oracle ADF

Oracle Application Development Framework. A range of technologies aimed at making **Java EE** application development faster and simpler for developers while at the same time taking advantage of proven software patterns to ensure that the developed application is scalable, performant, and the like.

OAM

See **Oracle Access Manager (OAM)**.

OHS

See **Oracle HTTP Server (OHS)**.

OmniPortlet

A component of **Oracle WebCenter Portal** that enables you to inject portal-like capabilities, such as portlets, content integration, and customization, into your **Oracle ADF Faces** applications.

Oracle Access Manager (OAM)

Part of Oracle's enterprise class suite of products for identity management and security, Oracle Access Manager provides a wide range of identity administration and security functions, including several single sign-on options for Spaces and Framework applications. OAM is the recommended single sign-on solution for Oracle WebCenter Portal 11g installations.

Oracle ADF Faces

Oracle **Oracle ADF** Faces is a rich set of user interface components based on the new **JavaServer Faces** JSR (JSR 127). Oracle ADF Faces provide various user interface components with built-in functionality, such as data tables, hierarchical tables, and color and date pickers, that can be customized and reused in an application.

Oracle WebCenter Content: Content Server

A content repository for building secure business libraries with check in and check out, revision control, and automated publishing in web-ready formats. Current information is available to authorized users anytime, anywhere.

Oracle WebCenter: Content Server is a component of Oracle WebCenter Content.

Oracle WebCenter Portal's Discussions Server

Backend discussions server for the Discussions and Announcements services.

Oracle Enterprise Manager

A component that enables administrators to manage Oracle Fusion Middleware services through a single environment. The Fusion Middleware administrator uses Enterprise Manager to configure, manage, and monitor Framework applications.

Oracle HTTP Server (OHS)

Software that processes web transactions that use the Hypertext Transfer Protocol (HTTP). Oracle uses HTTP software developed by the Apache Group.

Oracle Internet Directory

Oracle's LDAP V3 compliant LDAP server. It is used as a repository for provisioning users and groups. By default, the **Oracle Single Sign-On (OSSO)** authenticates user credentials against Oracle Internet Directory information about dispersed users and network resources. Oracle Internet Directory combines LDAP version 3 with the high performance, scalability, robustness, and availability of the Oracle database.

Oracle JDeveloper

Oracle JDeveloper is an integrated development environment (**IDE**) for building applications and web services using the latest industry standards for Java, XML, and SQL. Developers can use Oracle JDeveloper to create Java portlets.

Oracle JSF Portlet Bridge

Based on and conforming to JSR 329, the Oracle JSF Portlet Bridge enables application developers to expose a JSF application or task flow as a JSR 286 portlet for consumption in another application.

Oracle Metadata Services

See **MDS**.

Oracle Portal

A component used for the development, deployment, administration, and configuration of enterprise class **portal**s. Oracle Portal incorporates a portal building framework with self-service publishing features to enable you to create and manage information accessed within your portal.

See also **Oracle WebLogic Portal**.

Oracle Secure Enterprise Search (SES)

Provides easy-to-use search for public and secure data, with unified ranking results. With Framework applications, Oracle SES is set as the default and preferred search platform.

With Spaces applications, WebCenter Portal's internal live search adapters are set as the default search platform; however, large-scale implementations should be configured to use Oracle SES for best performance.

Oracle Single Sign-On (OSSO)

A component that enables users to log in to all features of the Oracle Fusion Middleware product suite, and to other web applications, using a single user name and password.

Oracle WebCenter Content's Site Studio

A powerful, flexible web development application suite that offers a comprehensive approach to designing, building, and maintaining enterprise-scale web sites. Site Studio uses Oracle WebCenter Content: Content Server as the main repository for a web site.

In WebCenter, Content Presenter integrates with Site Studio to allow you to create, access, edit, and display Site Studio contributor data files in either a Site Studio region template or a custom Content Presenter display template.

Oracle SOA Suite

A middleware component of Oracle Fusion Middleware. Oracle SOA Suite enables services to be created, managed, and orchestrated into SOA composite applications. Composites enable you to easily assemble multiple technology components into one SOA composite application. Oracle SOA Suite plugs into heterogeneous infrastructures and enables enterprises to incrementally adopt SOA.

Oracle Technology Network

See **OTN**.

Oracle WebCenter Content

Provides a flexible, secure, centralized, web-based repository that manages all phases of the content life cycle: from creation and approval to publishing, searching, expiration, and archival or disposition. It enables contributors to easily contribute content from native desktop applications, efficiently manage business content through rich library services, and securely access that content anywhere using a web browser. All content, regardless of content type, is stored in the web repository or database for management, reuse and access.

Oracle WebCenter Portal

A suite of services that enables you to build Oracle WebCenter Portal applications. Oracle WebCenter Portal reduces the front-end labor historically required to bring necessary business components to the user by capitalizing on the notion of Service Oriented Architecture (SOA). The suite includes a wide range of plug-and-play products, tools, and services that make it easy to build the applications your users need.

Oracle WebCenter Portal: Framework

A set of features provided by **Oracle WebCenter Portal** that augments the Java Server Faces (JSF) environment by providing additional integration and runtime customization options. It integrates capabilities historically included in portal

products, such as site navigation, page hierarchies, portlets, customization, personalization, and integration, directly into the fabric of the JSF environment. This eliminates artificial barriers for the user and provides the foundation for developing context-rich applications. You can selectively add only desired Oracle WebCenter Portal components or services to your framework application.

Oracle WebCenter Portal's Pagelet Producer

Provides a collection of useful tools and features that facilitate dynamic pagelet development.

Oracle WebCenter Portal: Services

A collection of Web 2.0 services that expose social networking and personal productivity features through various services.

- Activity Graph
- Announcements service
- Analytics service
- Discussions service
- Documents service
- Events service
- Instant Messaging and Presence service
- Links service
- Lists service
- Mail service
- Notes service
- Notifications service
- People Connections service
- Personalization for WebCenter Portal
- Polls service
- Recent Activities service
- RSS service
- Search service
- Tags service
- Worklist service

Oracle WebCenter Portal: Spaces

A WebCenter Portal application built using JSF, Oracle ADF, WebCenter Portal: Framework, WebCenter Portal: Services, and Composer. In the production documentation, it is commonly referred as **Spaces application (Spaces)**.

Oracle WebLogic Server Administration Console

A browser-based, graphical user interface to manage a WebLogic Server domain. Use to:

- Configure, start, and stop WebLogic Server instances
- Configure WebLogic Server clusters

- Configure WebLogic Server services, such as database connectivity (JDBC) and messaging (JMS)
- Configure security parameters, including creating and managing users, groups, and roles
- Configure and deploy your applications
- Monitor server and application performance
- View server and domain log files
- View application deployment descriptors
- Edit selected run-time application deployment descriptor elements

OTN

Oracle Technology Network. The online Oracle technical community that provides a variety of technical resources for building Oracle-based applications. You can access OTN at `http://www.oracle.com/technetwork`.

Oracle WebLogic Portal

WebLogic Portal enables you to provide a user interface to integrate your applications. WebLogic Portal lets you surface application data and functionality from heterogeneous environments into an integrated, dynamic, and customizable web-based portal user interface that can simultaneously support multiple devices. In addition to a powerful portal framework and its J2EE security foundation, WebLogic Portal provides many business services, such as content management, communities, personalization, search, and user management. WebLogic Portal provides a virtual content repository that lets you federate external content management systems into a single management interface. You can then build portals using content in those external resource. WebLogic Portal also provides a WLP content repository for creating and managing content.

Page Composer

See **Composer**.

Page Customizable component

A component provided in the **Composer** tag library that defines the editable area of a page at runtime. Within this area, users can edit properties for a component, add content to the page, arrange content, and so on.

page hierarchy

A model that associates pages in a parent-child relationship, where any page can have one or more sub pages. This parent-child model not only helps define the overall structure of the portal, but also allows child pages to inherit the security policies from their parent.

page parameter

A parameter associated with a page that can be used to store values that can then be passed to the components on the page. It also enables your page to take values through its URL. Page parameters are defined using the `<parameter>` tag at the top of your `PageDef.xml`. You can bind page parameters to your **page variable**s.

Page Properties

A dialog, accessed from **Composer**, that provides access to a page's display options, security settings, and parameters.

page scheme

Determines the background image used in the page. The Spaces application provides several default page schemes and an option for specifying a custom page scheme.

page style

Determines the initial page structure, for example one column or two column. Some default page styles also include the task flows, components, and page properties useful for a particular purpose. For example, a page created using the Text page style includes a Text layout component.

page template

Lets you specify view elements that you intend to be common to all of your pages. A page template file is a JSPX file that includes ADF layout components and other elements. Typically, page templates define a page layout, with headers, footers, and content areas. In addition, the page template usually specifies the positioning and style of the navigation UI for your pages.

page variable

A variable that binds your public portlet parameter to the page. Page variables are defined within the `<variableIterator>` of your `PageDef.xml`. One page variable can be bound to multiple public portlet parameters.

pagelet

A **portlet** that can be used in any web page (not just within a portal).

Panel Customizable component

A component provided in the **Composer** tag library that provides a container region for a group of Oracle ADF components and portlets that are customizable at runtime. Any Show Detail Frame components and portlets added as child components to a Panel Customizable component can be moved or maximized with the Panel Customizable component.

parameter

A variable that controls the default behavior of task flow content and facilitates the wiring of a task flow to page parameters and page definition variables.

parent component

A component that contains other components, such as a Box layout component that contains task flows. The Box is the parent component of the task flows. In contrast, the task flows are the Box's child components.

See also **Child Components**.

participant

A Spaces user who can manipulate the content of a space. A participant can upload and share documents, initiate and take part in chats with other members, create discussion topics, create new or view existing lists.

PDK-Java

Java Portlet Developer Kit. The development framework used to build and integrate web content and applications with **Oracle WebCenter Portal**. It includes toolkits, samples, and technical articles that help make portal development simple. You can take existing Java **servlet**s, **JSP**s, URL-accessible content and web services and turn

them into **portlet**s. It is typically used by external developers and vendors to create portlets and services.

People Connections service

A WebCenter Portal service that provides social networking tools for creating, interacting with, and tracking the activities of one's enterprise connections.

See also, **Activity Stream**, **Connections**, **Feedback**, **Message Board**, and **Profile**.

personalization

Dynamic changes to an application's behavior based on user context, facilitated by Personalization for WebCenter Portal.

See also **application customization** and **user customization**

Personalization for WebCenter Portal

A WebCenter Portal service that enables you to deliver content within your application to targeted application users based on selected criteria. Personalization for WebCenter Portal also provides a declarative means for specifying dynamic application flow.

personal page

A page created by a user in his or her Home space. Personal pages are viewable by other users only if specifically granted access by the user who created the page.

personal profile

A page that displays a user's personal information such as email address, phone number, office location, department, manager, direct reports, and so on.

See also, **Profile**.

Polls service

A WebCenter Portal service that enables you to create, edit, and take online polls on your application pages. Polls let you survey your audience (such as their opinions and their experience level) and check whether they can recall important information, and gather feedback on the efficacy of presentations.

portal

A common interface (that is, a web page) that provides a personalized, single point of interaction with web-based applications and information relevant to individual users or class of users.

portal application template

See **application templates**.

Portal Developer Kit

See **PDK-Java**.

portlet

A reusable web component that can draw content from many different sources. Portlets can display excerpts of other web sites, generate summaries of key information, perform searches, and access assembled collections of information from a variety of data sources. Because different portlets can be placed on a common page, the user receives a single-source experience, even though the content may be derived from multiple sources. Portlet resources include the many prebuilt portlets available

out of the box and programmatic portlets built through WebCenter's JSR 286, PDK-Java Portlet wizards, and other portlet building tools.

portlet event

A JSR 286 feature that allows portlets to react to actions or state changes not directly related to an interaction of the user with the portlet.

portlet filter

A JSR 286 feature that allows on-the-fly transformations of information in both the request to and the responses from a portlet. A portlet filter is a reusable piece of code that can transform the content of portlet requests and portlet responses.

portlet mode

The ways by which a **portlet** can be called to display information. These methods include:

- **Shared Screen mode** or **View mode**
- **Edit mode** or **Edit Defaults mode**
- **Customize mode**
- **Help mode**
- **About mode**
- **Full Screen mode (Portlets)** or **Show Details Page mode**

Portlet Producer Application template

An application template, provided by JDeveloper, for creating an application with the recommended projects and technology scopes required for developing portlets. The Portlet Producer Application template consists of a single project scoped for portlet creation (Portlets).

See also **Framework application template** and **producer**.

predefined page

A page created by the Spaces application to perform a specific function. Examples of predefined pages include, Welcome pages, Search pages, and Documents pages.

Predeployment Tool

A utility for **Framework application**s that assists you in configuring your target system with the new producer registrations you have added to your application in Oracle JDeveloper. You must run this utility before deploying your application. You can also use this utility after deployment to migrate metadata from stage to production, for example, to export and import your customizations. This tool also enables you to define the **MDS** repository location to allow run-time customizations to be migrated.

pretty URL

A shortened version of a page's URL that hides the complexity of the real web address.

private parameter

A portlet parameter that is known only to the portlet itself and has no connection to the page on which the portlet resides.

Contrast with **public parameter**.

producer

A communication link between portlet consumers (such as a **Framework application** or a **portal**). When a consumer application renders a portlet, it calls the producer of that portlet, which in turn executes the portlet and returns the results in the form of portlet content. A producer can contain one or more portlets. A portlet can be contained by only one producer.

Oracle WebCenter Portal supports two types of producers:

- Oracle **PDK-Java** producers: Deployed to a **Java EE** application server, which is often remote and communicates through Simple Object Access Protocol (SOAP) over HTTP.
- **Web Services for Remote Portlets** (WSRP): A web services standard that enables the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as **JSR 286**, .NET, Perl) and any WSRP portal. A portlet (regardless of language) deployed to a WSRP-enabled container can be rendered in any application that supports this standard.

Profile

In the **People Connections service**, a feature for viewing and managing information about yourself, such as your contact information, manager, and direct reports, and for viewing this information about other application users.

programmatic portlets

Portlets constructed in a non-declarative manner using **API**s. Also referred to as *hand-* or *manually-coded* portlets.

public render parameters

A JSR 286 feature that enables portlets to share parameter values, allowing a form of interportlet communication.

public space

A space that is available to all users, even those who are not logged in to the Spaces application.

public page

A page within the Spaces application that is available to all users, even those who are not logged in to the application.

public parameter

A portlet parameter that is known to the page and bound to it by way of a page variable.

Contrast with **private parameter**.

public user

A user who can access, but is not logged into, a WebCenter Portal application. A public user can view any page that has been marked as public, but cannot personalize or edit any content, or view pages that have any form of access control.

Contrast with **authenticated user**.

Recent Activities service

A WebCenter Portal service that provides a means of tracking recent activities in a Framework application.

recipe

A weighted list of similarity calculations. The weighting of each calculation determines its significance in deciding the overall recommendation score. Recommendations are ordered by their total recommendation score.

resize handle

A user interface element in a task flow chrome increasing or decreasing the height of the task flow.

resource

Items users can manage through the Resource Manager, like page templates, skins, task flows, navigations, and so on. Spaces categorizes resources under the following categories: Structure, Look and Layout, and Mashups.

Resource Action Handling framework

Enables services that expose custom resources to be viewed, searched, and tagged.

Resource Catalog

A catalog that provides a federated view of one or more otherwise unrelated repositories in a unified search and browse user interface. Resources are created and published in their source repository and are then exposed to the developer in JDeveloper's Resource Palette and to the end user in the Resource Catalog Viewer. Resource catalogs can contain layout components, Oracle ADF components, portlets, service task flows, and documents.

Resource Index

The starting point for accessing WebCenter Portal REST APIs. Sending a GET request to the Resource Index URI returns a list of links to entry points for all available services.

Resource Manager

Enables users with the appropriate privileges to continue developing the portal after the application has been deployed. Using the Resource Manager users can also download runtime portal resources (from Framework applications and Spaces) and import them into Oracle JDeveloper for further development. These resources can then be exported from JDeveloper and uploaded back into the deployed application.

resource type

Defines the type of resource that a WebCenter Portal REST API link identifies. Use resource types to determine the response bodies for GET requests and allowable request bodies for POST and PUT. Also use `resourceType` attributes on entities to uniquely identify their type.

REST APIs

Oracle WebCenter Portal provides a set of web-based REST (REpresentational State Transfer) APIs for retrieving and modifying server data dynamically from the client. REST APIs are available for **Discussions service**, **People Connections service**, and **Spaces application (Spaces)**.

Reverse Proxy Server

A server process that hides the physical location of internal servers by exposing the servers as a single public site. Requests to the public site are routed to the appropriate internal server.

Round-trip development

Round-trip development refers to features and techniques that allow you to retrieve resources from a deployed, runtime portal back to JDeveloper for maintenance or enhancement. After modifying a resource in JDeveloper, you can use the Resource Manager to upload the resource back to the deployed portal. WebCenter Portal's round-trip development features provide a simple, convenient way to modify portal resources without redeploying the entire application.

RSS reader

An RSS reader provided with the Spaces application that incorporates public news feeds from external sources onto application pages. This RSS reader is available only in Spaces, and not in Framework applications.

RSS service

A WebCenter Portal service that provides a means of publishing content from other services as news feeds. The RSS service supports both RSS 2.0 and Atom 1.0 formats.

Search service

A WebCenter Portal service that enables the discovery of information and people in an application, returning only the results users are authorized to see

Secure Enterprise Search

See **Oracle Secure Enterprise Search (SES)**.

secured application page

A page created by a user that has not been made available to public users.

Self-Registration page

A predefined page where users can register with the Spaces application, thus creating themselves an identity store login account. Administrators can customize certain aspects of this page.

Self-Subscription page

A predefined page where users can register to become members of a space. Moderators can customize certain aspects of this page.

service ID

In Expression Language, the string that identifies a particular service. For example, the string `oracle.webcenter.collab.announcement` is the service ID for the Announcements service.

A PDK-Java producer's unique identifier. PDK-Java enables you to deploy multiple producers under a single adapter servlet. Different producers are identified by their unique service IDs. A service ID is required only when a service ID/producer name is not appended to the URL endpoint.

Service Oriented Architecture

See **SOA**.

servlet

A Java program that usually runs on a **Web server**, extending the web server's functionality. HTTP servlets take client HTTP requests, generate dynamic content (such as through querying a database), and provide an HTTP response.

session language

A display language specified by the user that remains in effect for the life of the session cookie (from log on to log off). If the user clears browser cookies, the display language reverts to the user-level default language, if specified, then to the application-level default language set by the Spaces administrator. Set the session language in the Change Language pop-up, accessible from the Welcome page.

Shared Screen mode

A **portlet mode** that renders the body of the portlet and enables you to display a portlet on a page that can contain other portlets. Every portlet must have at least a Shared Screen mode.

See also **View mode**.

Show Detail Frame component

A component provided in the **Composer** tag library that renders a border or chrome around the child component. It provides a header with an Actions menu and thereby providers user interface (UI) controls to customize the display of the child component. However, to customize the display of the child component, the Show Detail Frame component must be included inside a Panel Customizable component.

Show Details Page mode

A **portlet mode** that provides full-browser display of the portlet. For example, a portlet in **Show Page mode** could be limited to displaying only the ten most recently submitted expense reports, while the same portlet in Show Details Page mode could show all submissions.

Contrast with **Show Page mode**.

show modes

Types of **portlet mode**s encompassing **Show Page mode** and **Show Details Page mode**.

Show Page mode

A **portlet mode** that provides a smaller portlet display to allow space for additional portlets and other objects in the browser window. For example, a portlet in Show Page mode could be limited to displaying only the ten most recently submitted expense reports, while the same portlet in Show Details Page mode could show all submissions.

Contrast with **Show Details Page mode**.

similarity calculation

Used by the Activity Graph service to provide a similarity score (a number between zero and one) that designates how similar two objects are to each other given a specific criterion. The weighting of each calculation determines its significance in deciding the overall recommendation score. Recommendations are ordered by their total recommendation score.

skin

A style sheet based on the CSS 3.0 syntax specified in one place for an entire application. Instead of providing a style sheet for each component, or inserting a style sheet on each page, you can create one skin for the entire application.

SOA

Service Oriented Architecture. A design methodology aimed at maximizing the reuse of application services.

Source view (JDeveloper)

A view, in **Oracle JDeveloper**, that provides a way to directly edit the source code of a file.

Source view (Spaces)

A view, in **Composer**, that provides a selectable structural representation of a page and its components.

See also Design view (Spaces).

space

A work area within Spaces that supports a group of people of any size that is organized around an area of interest or a common goal.

Spaces application (Spaces)

A web-based application that offers the very latest technology for social networking, communication, collaboration, and personal productivity. Spaces uses services and applications to bring everything together that users require to exchange ideas with others, keep track of personal and work-related tasks, interact with critical applications, and zero in on projects and interests; all within a single integrated environment.

space icon

An image displayed alongside space names on the Spaces page in My Spaces to help other users with identification and location.

space logo

An image displayed on the Home space page to provide a visual identity for the space. The Home space logos also display alongside the space name at the top of the page in Full Screen Mode.

space member

A user who is participating in a space. Members can be added or invited to a space, or they can subscribe to a space themselves if self-registration is enabled.

space owner

A user who initially created a space. The space owner is automatically also a moderator of the space.

space template

A starting point for creating a new space. Spaces includes several out-of-the-box templates to get you started, and you can create custom space templates using existing spaces as the basis.

space Unavailable page

A predefined page that displays when a space member tries to open a space that is temporarily offline. Moderators can customize this page.

Spaces application administrator

See administrator.

Spaces Switcher

A menu showing three areas: **Recent Spaces** lists up to ten recently accessed spaces, followed by spaces to which current user most recently gained access. **My Spaces** lists all spaces to which the current user has access, in alphabetical order. A list of links provides direct access from the menu to the Home space, the Spaces browser page, and the Create a Space dialog.

struts

A development framework for Java servlet applications based upon the **MVC** design paradigm.

style properties

Used to override the style information from the skin CSS to set specific instances of component display.

Tags service

A WebCenter Portal service that enables users to apply their own terms to application objects, making it possible to search for those objects using personally meaningful terms.

task flow

A set of ADF Controller activities, control flow rules, and managed beans that interact to allow a user to complete a task. Task flows provide a modular approach for defining control flow in an application. Instead of representing an application as a single JSF page flow, developers can break it up into a collection of reusable task flows.

task flow header

An area at the top of a task flow that displays the task flow name and various tools for interacting with the task flow.

template

See **space template**, **application templates**, and **custom display template**.

Text layout component

A layout component available through **Composer**. A rich text editor for providing static page text. For designers of Framework applications, this is the runtime equivalent of a Rich Text Editor component.

Unauthorized Access page

A predefined page that is shown when someone without access permission tries to open a page.

URL parameter

See **private parameter**.

user customization

Changes that affects only a user's own work space.

See also **application customization** and **personalization**

validation-based caching

A **caching** method that uses a validation check to determine if the cached item is still valid.

Contrast with **expiry-based caching**.

Virtual Content Repository

Virtual Content Repository (VCR) enables you to plug in multiple, heterogeneous content repositories.

viewer

Spaces users who can look at the information in a space but cannot contribute any of their own.

View mode

(**JSR 286** portlets only.) A **portlet mode** that enables you to display a JSR 286 portlet on a page that can contain other portlets. It is the only required mode for JSR 286 portlets.

See also **Shared Screen mode**.

WAR

Web application archive file. This file is used in deploying applications on a **Java EE** application server. WAR files encapsulate in a single module all of the components necessary to run an application. WAR files typically contain an application's **servlet**, **JSP**, and **JSF JSP** components.

See also **EAR**.

Web 2.0

Technologies, such as wiki, RSS, and blogs, that enable the construction of highly interactive web applications.

See also **Oracle WebCenter Portal: Services**.

Web Application Archive file

See **WAR**.

Web Clipping portlet

A browser-based declarative tool that enables you to integrate any web application with your **Framework application**. It is designed to give you quick integration by leveraging the web application's existing user interface. You can drag and drop Web Clipping portlets onto a `*.jspx` page.

Web Page layout component

A layout component available through **Composer**. A means of embedding another web site, wiki, or blog within the context of a page which is created in the Spaces application. For designers of Framework applications, this is the equivalent of an Inline Frame component.

Web server

A program that delivers web pages.

Web Services for Remote Portlets

See **WSRP**.

WebCenter Portal Application

A WebCenter Portal application based on **Oracle WebCenter Portal: Framework** or **Oracle WebCenter Portal: Spaces**.

WebCenter Portal's Extension for Oracle JDeveloper

An extension available through the Oracle JDeveloper Update Wizard that installs the necessary libraries, templates, wizards, and dialogs needed to build and deploy **Framework application**s in **Oracle JDeveloper**.

WebCenter Portal: Framework

See **Oracle WebCenter Portal: Framework**.

WebCenter Portal systems administrator

See administrator.

WebLogic Server

See **WLS**.

Welcome page

There are two types of Welcome page:

- Public Welcome page: A predefined page that users encounter before logging in to the Spaces application.
- Personal Welcome page: A predefined page that introduces users to their Home space.

wiki page

A page that provides in-place editing using HTML or a simple mark-up language. Any user with sufficient privileges can add, revise, and remove wiki content.

WLS

WebLogic Server. A scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA).

See also **Integrated WLS**

WLST

WebLogic Scripting Tool. A command line tool for managing Oracle Fusion Middleware components, such as Oracle WebCenter Portal.

Worklist service

A WebCenter Portal service that provides access to notifications, alerts, and BPEL tasks assigned to the current user.

WSRP

Web Services for Remote Portlets (WSRP) is a web services standard that allows the plug-and-play of visual, user-facing web services with portals or other intermediary web applications. Being a standard, WSRP enables interoperability between a standards-enabled container based on a particular language (such as JSR 286, .NET,

Perl) and any WSRP portal. A portlet (regardless of language) deployed to a WSRP-enabled container can be rendered on any portal that supports this standard.

XSL

Extensible Stylesheet Language (XSL) is the language used within style sheets to transform or render XML documents.

Index

Symbols

_followEnforced property, 40-3
_objectExtHandlerEnabled property, 40-3
_objects property, 40-3
_objectSecured property, 40-3

A

AboutPage.jsp file, A-8
accessibility
  at runtime, 2-17
  features, 2-16
  of HTML, 2-16
  portlets, 2-17
Accordion View display template, 31-25
actions (Activity Graph)
  about, 46-3
  defaults for WebCenter Portal, 46-3
  extending, 46-26
actionsCategory element, B-40
ACTIVITIES database connection, 47-5
Activity Graph
  Activity Graph provider, 67-5
  Personalization, 67-5
Activity Graph provider, 67-19
Activity Graph service
  about, 46-1
  actions
    about, 46-3
    defaults for WebCenter Portal, 46-3
    extending, 46-26
  Activity Rank, 46-2
  connections, 46-10
  data control, 46-18
  extending, 46-24
  node classes
    about, 46-2
    defaults for WebCenter Portal, 46-2
    extending, 46-25
  prerequisites, 46-7
  QRPPs
    about, 46-7
    defaults for WebCenter Portal, 46-7
    registering, 46-32
  rank calculations, extending, 46-29
  registering activity providers, 46-31
  REST APIs, 46-19
  similarity calculations
    about, 46-5
    defaults for WebCenter Portal, 46-5
    extending, 46-28
  task flows
    adding to a page, 46-14
    Recommended Connections, 46-7, 46-11
    Similar Items, 46-7, 46-13
    Similar Spaces, 46-7, 46-12
Activity Stream
  about, 38-2
  Advanced Query option, 40-12
  compared to Recent Activities service, 38-3
  REST APIs, 42-1, 54-22
  supported site-level settings, 39-5
  task flows
    Activity Stream, 38-4
    Activity Stream - Quick View, 38-4
  tracked activities, 38-2
Activity Stream - Quick View task flow
  about, 38-4
  parameters
    _objectExtHandlerEnabled, 40-3
    _objectSecured, 40-3
    hideConfigure, 40-6
    hideFooter, 40-7
    hideHeader, 40-7
    pageSize, 40-9
    profileUserId, 40-9
Activity Stream task flow
  about, 38-4
  parameters
    _followEnforced, 40-3
    _objectExtHandlerEnabled, 40-3
    _objects, 40-3
    _objectSecured, 40-3
    advancedQuery, 40-3
    enableContextInfo, 40-6
    hideComments, 40-6
    hideConfigure, 40-6
    hideInlinePreview, 40-7
    hideLike, 40-7
    pageSize, 40-9
    pagination, 40-9

profileOnly, 40-9
resourceId, 40-10
serviceCategories, 40-10
spaces, 40-11
adapters
Documents service, 30-5
Search service, 45-22
addMember API, 57-16
addon-config element, B-29
addon-panels element, B-38
add-ons
about, 21-2
creating, 21-4
Customization Manager, 21-9
excluding, 21-9
registering in adf-config.xml, 21-8
registering in extension file, 21-7
rendering selectively, 21-10
address portable contact type, 54-15
ADF data controls, 1-9
ADF security
configuring, 69-3
ADF, about, 2-10
adf-config.xml, 21-4, B-36, H-7
adfp:portlet tag
appearance attributes
allModesSharedScreen attribute, 64-26
background attribute, 64-27
displayActions attribute, 64-27
displayHeader attribute, 64-27
displayScrollBar attribute, 64-26
displayShadow attribute, 64-27
expansionMode attribute, 64-26
rendered attribute, 64-27
renderPortletInIFrame attribute, 64-26
shortDesc attribute, 64-27
showMinimizeAction attribute, 64-28
showMoveAction attribute, 64-28
showRemoveAction attribute, 64-28
showResizer attribute, 64-28
behavior attributes
partialTriggers attribute, 64-29
submitUrlParameters attribute, 64-29
binding attribute, 64-31
common attributes
height attribute, 64-25
icon attribute, 64-25
id attribute, 64-24
partialTriggers attribute, 64-25
title attribute, 64-24
width attribute, 64-24
customization attributes
customizationAllowed attribute, 64-31
customizationAllowedBy attribute, 64-31
example, 64-19
iframeDtd attribute, 64-32
placement, 64-20
portlet mode attributes
isAboutModeAvailable attribute, 64-29
isConfigModeAvailable attribute, 64-29
isCustomizeModeAvailable attribute, 64-29
isDetailModeAvailable attribute, 64-29
isHelpModeAvailable attribute, 64-30
isNormalModeAvailable attribute, 64-30
isPersonalizeModeAvailable attribute, 64-30
isPreviewModeAvailable attribute, 64-30
isPrintModeAvailable attribute, 64-30
required attributes, 64-21
setting attribute values, 64-21
style attributes
contentStyle attribute, 64-31
inlineStyle attribute, 64-31
usage, 64-19
adfph:portlet tag, 64-19
adf-settings.xml, B-47
adftemplate property, 19-9
administration console, portal, 9-20
Advanced Query option, 40-12
advancedQuery property, 40-3
allModesSharedScreen portlet tag attribute, 64-26
Analytics Collector
about, 47-2
setting properties, 47-6
setting up a connection, 47-7
Analytics database (ACTIVITIES)
about, 47-2
setting up a connection, 47-6
Analytics service
about, 47-1
building custom reports, 47-16
connection
connecting to Analytics Collector, 47-7
connecting to Analytics database, 47-6
setting up a connection, 47-5
enabling task flow customization, 47-9
setting Analytics Collector properties, 47-6
setting up security, 47-15
task flows
adding Analytics task flows, 47-9
Blog Metrics, 47-2
Discussion Metrics, 47-2
Document Metrics, 47-2
Login Metrics, 47-2
Page Traffic, 47-2
Portlet Instance Response Time, 47-2
Portlet Instance Traffic, 47-2
Portlet Response Time, 47-2
Portlet Traffic, 47-2
Search Metrics, 47-2
WebCenter Portal Traffic, 47-2
Wiki Metrics, 47-2
Announcements - Quick View task flow
about, 33-8
parameters
expandAllAnnouncements, 33-10
expandedAnnouncements, 33-10
freeFlowView, 33-10
hideToolbar, 33-10
navigateToAnnouncementViewer, 33-10
pageSize, 33-10

parentId, 33-10
truncateAt, 33-10
visibleAnnouncements, 33-10
Announcements service
about, 33-1
connection, 33-3
customizing views, 33-11, 33-13
requirements, 33-2
security, 33-11
task flows
Announcements, 33-8
Announcements - Quick View, 33-8
Announcements task flow
about, 33-8
parentId parameter, 33-10
anonymous-role (Framework application), 69-8
Apache MyFacesTrinidad, 30-5
Apache Struts, creating portlets, 61-61
APIs
back-end APIs, 7-2
data controls, 7-2
Discussions service, 34-15
expression language expressions (ELs), G-1 to G-19
Java, 7-2
Notifications Java APIs, 49-14
Page service, 19-18, 19-33
People Connections service Java APIs, 41-10
People Connections service REST APIs, 42-1 to 42-30
REST, 7-2, 42-1 to 42-30
Search service, 45-17
Tags service, 44-9
Web Clipping Transport, 66-2, 66-7
WebCenter Portal services, 7-2
WebCenter Portal: Spaces, 57-4
application customization, 18-2
Application Development Framework
See ADF
application links, 61-8
Application Resources panel, adding files and folders to page, 31-15
application resources, EL expressions relating to, G-3
application roles
Administrator, 69-2
AppConnectionManager, 69-2
AppConnectionViewer, 69-2
creating, 69-2
Framework application, 69-2
application sources folder, 9-18
applications
EL expressions relating to, G-1
exposing applications as JSF portlets, 59-2
exposing in WebCenter Portal: Spaces, 57-53
Archives task flow
parameters
resourceID, 32-12
automatic portlet wiring, 64-20
AutoVue pane, 31-37
AutoVue task flow
about, 31-4
resourceId parameter, 31-41

B

BACK_LINK URL, 61-45
background portlet tag attribute, 64-27
Banner task flow
parameters
imageURL, 32-12
resourceID, 32-12
title, 32-12
best practices
About mode, 61-5
CSS for portlets, 61-3
Edit Defaults mode, 61-5
Edit mode, 61-4
Help mode, 61-5
HTML for portlets, 61-2
Java portlets, 61-1
portlet navigation, 61-6
Preview mode, 61-5
bidirectional links, 43-3
binding portlet tag attribute, 64-31
Blog Archives task flow, 32-6
parameters
resourceID, 32-12
Blog Banner task flow, 32-6
parameters
imageURL, 32-12
resourceID, 32-12
title, 32-12
Blog Digest task flow, 32-6
parameters
hideComments, 32-13
month, 32-13
pageSize, 32-13
resourceID, 32-13
year, 32-13
Blog Metrics task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
ResourceId, 47-14
Blog Recent Posts task flow
parameters
recentPostListSize, 32-13
resourceID, 32-13
blogs
about, 32-1
adding blog task flows, 32-9
enabling the blog functionality, 32-7
establishing folder for blogs, 32-8
integrating, 32-5
modifying task flow parameters, 32-11
requirements, 32-1

task flows, 32-5
Blogs task flow, 32-7
parameters
hideComments, 32-14
pageSize, 32-14
resourceID, 32-14
BPEL server, configuring notifications, 49-6
bridge
See Oracle JSF Portlet Bridge
build environment, 10-6
builds, nightly, 10-5, 10-10
Bulleted View display template, 31-25
Bulleted with Folder Label View display template, 31-25

C

caching
expiry-based caching, 58-21
about, 61-32, 61-59
in JSR 286 portlets, 61-32
in PDK-Java portlets, 61-60
Java portlets, 58-22
JSR 286 portlets, 61-31 to 61-34
OmniPortlet, 58-22
PDK-Java portlets, 61-58 to 61-61
portlets, 58-21
validation-based caching, 58-21
about, 61-32, 61-59
in JSR 286 portlets, 61-33
in PDK-Java portlets, 61-61
Web Clipping, 58-22
Calendar Main View task flow, 52-7
Calendar Mini View task flow, 52-7
capabilities attribute, 54-8
Carousel View display template, 31-25
catalog definition
attributes, C-10
schema, C-2
catalog definition attributes
attribute, C-17
attributes, C-17
catalogDefinition, C-10
component, C-14
customContent, C-16
customFolder, C-15
folder, C-11
includeCatalog, C-13
parameters, C-19
schema, C-19
url, C-12
catalog registry, 9-4, 9-13
catalogDefinition, C-10
catalogs
about, 9-4
catalogs, resource, 9-13
cateogryId property, 34-10
Change Mode Button
about, 18-48
adding to a page, 20-4
properties, B-4
Change Mode Link
about, 18-48
adding to a page, 20-4
properties, B-4
CHANGE_PERSISTENCE context parameter, B-48
character sets
PDK-Java portlets, 60-17
Web Clipping, 66-9
classURN property, 46-16
classURNRestrictions property, 46-17
CMIS
security, 54-13
CMIS provider, 67-5, 67-16
Comments pane, 31-37, 31-39
Composer
about, 18-1
adding Box components, 18-13
adding content, 18-8
add-ons
about, 21-2
creating, 21-4
excluding, 21-9
rendering selectively, 21-10
application customization, 18-2
applying styles to components, 22-25
changing layouts, 18-28
components, 18-46, B-1
Change Mode Button, 18-48, B-4
Change Mode Link, 18-48, B-4
Custom Action, B-23
Layout Customizable, 18-49, B-6
Page Customizable, 18-47, B-2
Panel Customizable, 18-48, B-10
Show Detail Frame, 18-49, B-16
Show Property, B-26
concurrency behavior, 23-6
configuration files, 21-4, B-29
adf-config.xml, B-36
adf-settings.xml, B-47
pe_ext.xml, B-29
web.xml, B-47
configuring a keyboard shortcut, 22-21
configuring custom LOVs, 21-22
creating labels to save customizations, 18-32
Custom Action component, B-23
custom property panels
about, 21-2
creating, 21-11
excluding, 21-16
overriding, 21-15
registering, 21-12
Customization Manager, 18-33
customizing the toolbar, 21-53
cut and paste options, 18-40
default add-ons, B-48
default configurations, 20-12
default property panels, B-48
deleting components, 18-28
design considerations, 20-13

design time at runtime, 18-1
Design view, 18-7
designing editable pages, 20-1
disabling
  disabling runtime editing of pages, B-6
  disabling Source view, 22-23
  disabling task flow zoom, 22-24
displaying source view by default, 22-22
drop handlers, 21-3, 21-42
  creating, 21-42
  registering, 21-45
Edit mode capabilities, 18-6
editing
  editing component properties, 18-17
  editing page properties, 18-29
  editing resource strings, 18-21
  editing task flow content, 18-41
enabling runtime customizations, 20-3
error handling, 18-32
events
  about event handlers, 21-2
  configuring, 21-37
  events supporting event handler registration, 21-38
  registering, 21-40
extending framework, 21-1
extending the expression builder, 21-20
global style selectors, B-53
hiding page components
  Design view, 18-27
  Source view, 18-45
managing customizations, 18-33
MDS configuration
  about, 23-1
  adding customization layers, 23-8
  MDS customization restrictions, 18-54
overriding
  component customizations, 18-20
  default security behavior, 24-1
  default security policies, 24-23
page privileges mapping, 18-51
page-level security, 18-51
persistence change manager
  about, 22-27
  configuring, 22-29
property filters, 21-3
  defining, 21-48
  removing, 21-49
rearranging components using cut and paste options, 18-40
rearranging content, 18-16
resetting component customizations, 18-20
resetting customizations, 18-32
resetting page customizations, 18-30
resetting task flow customizations, 18-43
resource string editor
  about, 22-31
  configuring, 22-30
sandbox, 23-2
  destroying stale sandboxes, 23-7
  disabling, 23-7
  enabling, 23-3
  saving customizations, 23-5
  security
    action-level restrictions, 24-16
    attribute-level restrictions, 24-14
    component-level restrictions, 24-1
    customization policies, 24-1
    customizationAllowed attribute, 24-13
    security considerations, 18-50
    task flow-level restrictions, 24-21
    task flow-level security, 18-53
  showing page components
    Design view, 18-27
    Source view, 18-45
  Source view, 18-37
  user customization, 18-2
  View mode capabilities, 18-3
ComposerSessionOptionsFactory class, 23-14
Conductor, 67-2
  REST APIs, 68-96
config.properties (WebCenter Spaces), 56-8
configuration
  ADF security, 69-3
  custom validators, 1-10
  faces-config.xml file, 1-10
  JSF servlet settings, 1-9
  managed beans, 1-10
  navigation rules, 1-10
  servlet mapping, 1-9
  web.xml file, 1-9
configuring basic authentication, 69-26
connectionListName property, 40-4
connectionName property
  Document Explorer task flow, 31-26
  Document List Viewer task flow, 31-28
  Document Manager task flow, 31-30
  Document Navigator task flow, 31-32
  Folder Viewer task flow, 31-33
  Recent Documents task flow, 31-35
Connections
  about, 38-5
  accepting invitations automatically, 39-11
  example site-level setting, 39-11
  REST APIs, 42-10
  supported site-level settings, 39-6
  task flows
    Connections, 38-5
    Connections - Card, 38-5
    Connections - Quick View, 38-6
connections
  about, 1-7
  accessing connection wizards, 1-7
  Announcements service, 33-3
  data source repositories, 1-7
  Discussions service, 34-3
  Documents service, 30-2
  Events service, 52-4
  files affected, 4-5
  IMP service, 35-2

Links service, 43-4
locations for creating connections, 1-7
Mail service, 36-5
Oracle SES (Oracle Secure Enterprise Search), 45-6
People Connections service, 39-1
Polls service, 37-3
portlet producers, 1-7
Recent Activities service, 50-3
RSS service, 53-3
Tags service, 44-3
Connections - Card task flow
about, 38-5
parameters
connectionListName, 40-4
display_maxConnections, 40-4
display_numberOfColumns, 40-4
display_numberOfRows, 40-4
display_profileFormat, 40-5
display_removalAllowed, 40-5
display_sortedBy, 40-5
filterPattern, 40-6
hideFooter, 40-7
profileLaunchedOnSelection, 40-9
userId, 40-11
Connections - Quick View task flow
about, 38-6
parameters
display_maxConnections, 40-4
display_numberOfColumns, 40-4
display_numberOfRows, 40-4
hideFooter, 40-7
hideHeader, 40-7
userId, 40-11
Connections task flow
about, 38-5
parameters
currentView, 40-4
numberOfRows_list, 40-8
resourceId, 40-10
constructResourceURL method, 61-8
container runtime options
setting at application level, 61-19
supported, 61-12
content
adding to a page, 31-12
introduction to integrating and publishing, 26-1
methods to integrate, 26-1
content data controls
about, 28-1
comparison overview, 26-2
configuring, 28-5
creating, 28-5
default attributes, 28-1
editing, 28-8
methods
about, 28-1
advancedSearch method, 28-3
getAttributes method, 28-4
getItems method, 28-2
getURI method, 28-4
search method, 28-2
parameters, 28-1
securing, 28-9
use case examples, 28-9
display file names as hyperlinks, 28-30
enable advanced search, 28-34
enable simple search, 28-33
publish as tables, 28-17
publish as tree, 28-27
publish content as links, 28-10
content integration
about, 2-4, 26-1
methods of integrating content, 26-1
content management REST APIs, 26-2
Content Presenter
comparison overview, 26-2
presenting Personalization content, 67-27
Content Presenter display templates
Accordion View, 31-25
Bulleted View, 31-25
Bulleted with Folder Label View, 31-25
Carousel View, 31-25
Default Document Details View, 31-24
Default List Item View, 31-24
Default View, 31-24
Icon View, 31-25
List View, 31-25
List with Details Panel View, 31-25
Sortable Table View, 31-25
Tabbed View, 31-25
Content Presenter task flow
about, 31-1
display templates, 29-2, 31-24
EL expressions, 29-10
in page templates, 12-15
multiple content item display templates, 29-8
overview, 29-1
parameters, 31-22
datasource, 31-22
datasourceType, 31-23
regionTemplate, 31-23
taskFlowInstId, 31-24
templateCategory, 31-24
templateView, 31-24
runtime behavior, 29-24
single content item display templates, 29-4
templates, generating valid list, 29-24
content repositories
adding content to a page, 31-12
Documents service, 30-1
uploading files, 30-4
content repository connections
connection type
Content Server-based, 27-2
file system-based, 27-15
Oracle Portal-based, 27-8
Oracle WebCenter adapter for SharePoint-based, 27-11
editing application-specific connections, 27-23

editing common connections, 27-21
external application, 27-17
identity propagation, 27-17
Oracle WebCenter adapter for SharePoint-based, 27-12
using existing repository connections, 27-23
Content Server
configuration parameters, 27-3
setting up connection, 27-2
content type
definition, 29-2, 29-15
list of properties, 29-15
properties, 29-15
contentListTemplate tag, 29-9
contentListTemplateDef tag, 29-8
contentStyle portlet tag attribute, 64-31
contentTemplate tag, 29-6, 29-9
contentTemplateDef tag, 29-5
cookies, Web Clipping, 66-8
createdAfter property, 31-28
createdBefore property, 31-28
createGroupSpace API, 57-14
createGroupSpaceTemplate API, 57-15
creator property, 31-28
CROs
See container runtime options
CSS
creating a CSS file, 15-3
guidelines for portlets, 61-3
using with Web Clipping, 66-8
currentView property, 40-4
custComps-config, B-47
Custom Action component
about, B-23
properties, B-23
custom actions
adding to a Show Detail Frame, 22-9
adding to a task flow, 22-7
enabling at global level, 22-10
enabling at instance level, 22-9
task flows, 22-16
custom certificates, 69-36
custom portlet modes, adding, 61-10
custom property panels
about, 21-2
creating, 21-11
excluding, 21-16
overriding, 21-15
registering, 21-12
custom Resource Catalog, 16-4
customActions element, B-47
custom-actions element, B-41
customContent attribute, C-16
customFolder attribute, C-15
customizable page
adding content, 20-17
example, 20-14
template, 20-10
customizableComponentsSecurity element, B-40
customization layers, 23-8
Customization Manager, 18-33
about, 21-3
deleting customizations, 18-36
displaying in Composer, 21-9
downloading customizations, 18-35
enabling parameter support, 21-50
parameters, 21-50
promoting customizations from a saved label, 18-36
rendering, 21-9
uploading customizations, 18-35
customization policies, 24-1
customization restrictions, 24-4
customizationAllowed attribute, 24-13
customizationAllowed portlet tag attribute, 64-31
customizationAllowedBy portlet tag attribute, 64-31
customizations
about, 9-4
applying customization restrictions, 24-4
component properties, 18-17
editing page properties, 18-29
exporting for WSRP 2.0, 61-34
importing for WSRP 2.0, 61-34
migrating, 70-24
cutsomAttributes property, 45-11

D

data controls
about, 7-5
Activity Graph service, 46-18
adding, 41-1
ADF data controls, 1-9
Notification Subscription, 49-16
Page service, 19-1, 19-18
People Connections service, 41-1 to 41-10
People Connections Management data control, 41-3
Profile data control, 41-10
Search service, 45-21
using, 7-5
data providers, 67-3, 67-4
custom, 67-6
database connection
about, 7-9
creating, 7-9
setting up for People Connections service, 39-1
WebCenter schema, 37-3, 43-4, 44-3
database schemas, installing, 3-6
DataBindings.cpx, H-3
DataDirect JDBC drivers
installing, E-5
registering with OmniPortlet, E-5
datasource property, 31-22
datasourceType property, 31-23
Default Document Details View display template, 31-24
Default List Item View display template, 31-24
default navigation model, selecting, 11-25
Default View display template, 31-24

default-navigation-model.xml, 11-2
defaultTopicFetchSize property, 34-10
delegated administration, 9-4
deleteGroupSpace API, 57-15
deployment, 10-12
  migrating customizations, 70-24
  packaging credentials, 70-9
  packaging identity data, 70-10
  policy migration, 70-10
  using global data sources, 70-13
  using local data sources, 70-13
  WebLogic Managed Server
    provisioning, 70-12
deployment descriptor file, 1-9
depth property, 14-6
design considerations, 2-12
design time at runtime, 18-1
design time at runtime portlets, 58-14
DESIGN_LINK URL, 61-45
DesignWebCenterSpaces.jws, 56-1
destinationID property, 31-39, 31-40
destinationSelectionEnabled property, 31-40
developer, design considerations, 2-14
development techniques
  iterative development, 1-4, 9-17
  round-trip development, 1-8, 9-17
disableToolbar property, 34-10
Discussion Forums task flow
  about, 34-7
  parameters
    categoryId, 34-10
    defaultTopicFetchSize, 34-10
    doNotAllowSelectingPageSize, 34-10
    forumId, 34-10
    isCategoryView, 34-10
    showRecursiveForums, 34-10
Discussion Metrics task flow
  about, 47-2
  adding to a page, 47-9
  parameters
    analyticsReportTitle, 47-14
    analyticsResourceId, 47-14
    applicationName, 47-14
    maxDataPointsPerSeries, 47-14
discussions
  categories, 34-1, 34-11
  forums, 34-1, 34-11
    watching forums, 34-13
    watching topics, 34-13
Discussions - Popular Topics task flow
  about, 34-7
  parameters
    categoryId, 34-10
    disableToolbar, 34-10
    forumId, 34-10
Discussions - Quick View task flow
  about, 34-7
  categoryId parameter, 34-10
  parameters
    doNotShowMoreLink, 34-10
Discussions - Recent Topics task flow
  about, 34-7
  parameters
    categoryId, 34-10
    disableToolbar, 34-10
    forumId, 34-10
Discussions - Watched Forums task flow
  about, 34-7
  parameters
    categoryId, 34-10
    disableToolbar, 34-10
Discussions - Watched Topics task flow
  about, 34-7
  parameters
    categoryId, 34-10
    disableToolbar, 34-10
    forumId, 34-10
discussions server
  about, 34-1, 34-2
  credentials, 34-8
Discussions service
  about, 34-1
  adding to Framework applications, 34-6
  APIs, 34-15
  configuring with IMP service, 34-8
  configuring with Links service, 34-8
  connections, 34-3
  customizing views, 34-11
  requirements, 34-2
  security, 34-12
  task flows
    Discussion Forums, 34-7
    Discussions - Popular Topics, 34-7
    Discussions - Quick View, 34-7
    Discussions - Recent Topics, 34-7
    Discussions - Watched Forums, 34-7
    Discussions - Watches Topics, 34-7
  troubleshooting, 34-24
display templates
  at runtime, 29-24
  Content Presenter, 31-24
  multiple-item, 29-8
  performance considerations, 29-25
  single-item, 29-4
  testing, 29-22
  using, 29-18
display_maxConnections property, 40-4
display_numberOfColumns property, 40-4
display_numberOfRows property, 40-4
display_profileFormat property, 40-5
display_removalAllowed property, 40-5
display_sortedBy property, 40-5
displayActions portlet tag attribute, 64-27
displayCount property, 40-5
displayHeader portlet tag attribute, 64-27
displayMessageSize property, 40-5
displayScrollBar portlet tag attribute, 64-26
displayShadow portlet tag attribute, 64-27
document components, adding to a page, 31-10
Document Explorer task flow

about, 31-3
parameters, 31-25
connectionName, 31-26
featuresOff, 31-26
pageSize, 31-26
readOnly, 31-26
resourceId, 31-26
showDocuments, 31-27
showFolders, 31-27
startFolderPath, 31-27
treeNavCollapsed, 31-27
Document List Viewer task flow
about, 31-3
parameters, 31-27
connectionName, 31-28
createdAfter, 31-28
createdBefore, 31-28
creator, 31-28
lastModifiedAfter, 31-28
lastModifiedBefore, 31-28
lastModifier, 31-28
pageSize, 31-29
showFolders, 31-29
startFolderPath, 31-29
taskFlowInstId, 31-29
Document Manager task flow
about, 31-3
parameters, 31-30
connectionName, 31-30
featuresOff, 31-30
layout, 31-30
pageSize, 31-31
readOnly, 31-31
resourceId, 31-31
startFolderPath, 31-31
Document Metrics task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
Document Mini Properties task flow
about, 31-3
resourceId parameter, 31-38
Document Navigator task flow
about, 31-3
parameters, 31-32
connectionName, 31-32
featuresOff, 31-32
pageSize, 31-32
readOnly, 31-32
resourceId, 31-32
showDocuments, 31-32
startFolderPath, 31-33
Document Properties task flow
about, 31-3
parameters, 31-38
readOnly, 31-38
resourceId, 31-38
Document Upload task flow
about, 31-3
parameters, 31-40
destinationID, 31-40
destinationSelectionEnabled, 31-40
featuresOff, 31-40
Document Version History task flow
about, 31-4
parameters, 31-40
readOnly, 31-41
resourceId, 31-41
Document Viewer task flow
about, 31-3
parameters, 31-37
featuresOff, 31-37
initialSidebar, 31-37
readOnly, 31-37
resourceId, 31-37
Documents service
adapters, 30-5
adding files and folders to a page, 31-14
comparing methods of integrating content, 26-2
connections, 30-2
content repositories, 30-1
EL expressions relating to, G-14
integrating, 30-1
security, 30-2
task flows
about, 31-2
adding, 31-16
adding files and folders, 31-14
adding to page from Application Resources panel, 31-15
AutoVue, 31-4
Document Explorer, 31-3
Document List Viewer, 31-3
Document Manager, 31-3
Document Mini Properties, 31-3
Document Navigator, 31-3
Document Properties, 31-3
Document Upload, 31-3
Document Version History, 31-4
Document Viewer, 31-3
Folder Viewer, 31-3
Recent Documents, 31-3
Rich Text Editor, 31-3
using with other WebCenter services, 30-4
doNotAllowSelectingPageSize property, 34-10
doNotShowMoreLink property, 34-10
drop handlers, 21-42
about, 21-3
creating, 21-42
registering, 21-45
drop-handler element, B-33
drop-handlers element, B-33

E

E-Business Suite applications, 55-9

ecurity
overriding page-level security, 69-39
Edit mode, 18-3
EditDefaultsPage.jsp file, A-8
editMode property, 40-5
EditPage.jsp file, A-8
EL expressions, G-1 to G-19
application, G-1
application resources, G-3
Content Presenter task flow, 29-10
Documents service, G-14
general settings, G-2
navigation, 14-10, G-6
People Connections service, G-14
Profile, G-14
security, G-2
service IDs, G-13
services, G-12
WebCenter Personalization, G-17
elbuilder-config element, B-35
enableContextInfo property, 40-6
enableSecurity element, B-40
Enterprise Role Member task flows, 69-17
Security - Enterprise Role Members, 69-17
event-handler element, B-33
event-handlers element, B-33
Events service
about, 52-1
adding to a Framework application, 52-7
connection, 52-4
requirements, 52-2
security, 52-8
task flows
Calendar Main View, 52-7
Calendar Mini View, 52-7
troubleshooting, 52-14
events, portlets, 61-21
excludeObjectActions property, 46-17
expandAllAnnouncements property, 33-10
expandedAnnouncements property, 33-10
expansionMode portlet tag attribute, 64-26
expiry-based caching, 58-21
about, 61-32, 61-59
in JSR 286 portlets, 61-32
in PDK-Java portlets, 61-60
exporting
customizations for WSRP 2.0, 61-34
resources from JDeveloper, 17-17
expression builder, extending, 21-20
Expression Language
Personalization, 68-195
See EL expressions, 29-10
expression language expressions (ELs), G-1 to G-19
Ext, using with REST APIs, 54-24
extAppId property, 53-5
extension bundle, WebCenter Portal, 3-1
external applications, 69-27
adding external application task flows, 69-32
adding to portal navigation, 11-9
associating with a PDK-Java producer, 64-13
associating with a WSRP producer, 64-9
authenticating, 58-28
automated single sign-on, 69-30
deleting registration, 69-36
editing registration details, 69-35
PDK-Java portlets, 69-50
registering, 69-33
security, 69-33
Web Clipping, 58-7
WebCenter Portal services that support external applications, 7-12
external links, portlets, 61-8

F

faces-config.xml, 1-10, H-4
featuresOff property
Document Explorer task flow, 31-26
Document Manager task flow, 31-30
Document Navigator task flow, 31-32
Document Upload task flow, 31-40
Document Viewer task flow, 31-37
Folder Viewer task flow, 31-34
Rich Text Editor task flow, 31-39
Federated Portal Adapter
about, F-4
limitations, F-7
registering, F-4
troubleshooting, F-6
Feedback
about, 38-6
REST APIs, 42-23
supported site-level settings, 39-7
task flows
Feedback, 38-6
Feedback - Quick View, 38-7
Feedback - Quick View task flow
about, 38-7
parameters
displayCount, 40-5
displayMessageSize, 40-5
fromDate, 40-6
hideActions, 40-6
hideFooter, 40-7
hideGiven, 40-7
hideHeader, 40-7
toDate, 40-11
userId, 40-11
Feedback task flow
about, 38-6
parameters
displayCount, 40-5
fromDate, 40-6
hideGiven, 40-7
hideHeader, 40-7
resourceId, 40-10
toDate, 40-11
file extensions, 1-10
files, 1-10
about, A-2

actions affecting metadata files, 4-4
adding to a page, 31-12, 31-14
deployment descriptor file, 1-9
faces-config.xml file, 1-10
for Framework applications, A-1
index.html file, 3-6
Integrated WLS readme file, 3-3
managing, 4-4
organization of portal application files
in Application Navigator, 9-22
on filesystem, 9-24
provider.xml, E-11
uploading, 30-4
WAR files, 6-6
web.xml file, 1-9, 1-10
filter-config element, B-34
filterPattern property, 40-6
filters, Activity Stream and, 40-12
Folder Viewer task flow
about, 31-3
parameters, 31-33
connectionName, 31-33
featuresOff, 31-34
pageSize, 31-34
readOnly, 31-34
resourceId, 31-34
showFolders, 31-34
startFolderPath, 31-35
folder, adding to a page, 31-14
folders
adding to a page, 31-12
form tag, IFRAME, 64-33
forms
building with URL types, 61-46
usesUpload attribute, 38-10, 39-2
forumId property, 34-10
FPA (Federated Portal Adapter)
about, F-4
limitations, F-7
registering, F-4
troubleshooting, F-6
frame sets, Web Clipping, 66-8
Framework application
ADF security, 69-3
application roles, 69-2
automated security grants, 69-8
security, 69-2
Framework applications
adding content, 13-1
adding content from Oracle WebCenter Content: Content Server, 13-3
adding custom task flows, 13-3
adding data controls, 13-5
adding external applications, 13-5
adding pagelets, 13-4
adding pages, 13-1
adding portlets, 13-4
adding service task flows, 13-2
application-level deployment profile, 70-11
consuming portlets, 64-1
default navigation model, 11-2
deploying, 70-1
deployment profile, 70-10, 70-11
EAR file, 70-11
Enterprise Archive (EAR) file, 70-10
introduction, 9-1
testing, 70-1
Framework in WebCenter Portal, 9-3
freeFlowView property, 33-10
fromDate property, 40-6
function providers, 67-4, 67-6

G

getCreatedBy API, 57-20, 57-23
getCustomAttributes API, 57-20
getDescription API, 57-20, 57-23
getDiscoverableGroupSpaces API, 57-20
getDisplayName API, 57-20, 57-23
getGroupSpaceMetadata API, 57-20
getGroupSpaces API, 57-19
getGroupSpaceState API, 57-20
getGroupSpaceTemplateMetadata, 57-22
getGroupSpaceTemplates API, 57-22
getGroupSpaceURL API, 57-24
getGuid API, 57-20, 57-23
getIconURL API, 57-20, 57-23
getKeywords API, 57-20, 57-23
getLastUpdated API, 57-20
getLocalizedMessage API, 57-26
getLogoURL API, 57-20, 57-23
getMailingList API, 57-21
getMembers API, 57-21
getName API, 57-21, 57-23
getPublicGroupSpaces API, 57-19
getRoles API, 57-18
getWebCenterSpacesURL API, 57-23, 57-25
global data sources, 70-13
global-attribute-filter element, B-34
globalization support, Web Clipping, 66-9
groupSpace property, 31-36
groupSpaceReference data type, 54-14
GroupSpaceWSClient class, 57-12, 57-14
addMember API, 57-16
createGroupSpace API, 57-14
createGroupSpaceTemplate API, 57-15
deleteGroupSpace API, 57-15
getDiscoverableGroupSpaces API, 57-20
getGroupSpaceMetadata API, 57-20
getGroupSpaces API, 57-19
getGroupSpaceTemplateMetadata API, 57-22
getGroupSpaceTemplates API, 57-22
getGroupSpaceURL API, 57-24
getPublicGroupSpaces API, 57-19
getRoles API, 57-18
getWebCenterSpacesURL API, 57-23, 57-25
inviteMember API, 57-17
removeMember API, 57-17
setGroupSpacesWebServiceEndpoint API, 57-10
GroupSpaceWSContext class, 57-11

GroupSpaceWSException class, 57-26
getLocalizedMessage API, 57-26
printStackTrace API, 57-27
GroupSpaceWSMembers class, 57-16, 57-17
GroupSpaceWSMetadata class, 57-20, 57-23
getCreatedBy API, 57-20, 57-23
getCustomAttributes API, 57-20
getDescription API, 57-20, 57-23
getDisplayName API, 57-20, 57-23
getGroupSpaceState API, 57-20
getGuid API, 57-20, 57-23
getIconURL API, 57-20, 57-23
getKeywords API, 57-20, 57-23
getLastUpdated API, 57-20
getLogoURL API, 57-20, 57-23
getMailingList API, 57-21
getMembers API, 57-21
getName API, 57-21, 57-23
isDiscoverable API, 57-21
isPublic API, 57-21
guidelines
About mode, 61-5
CSS for portlets, 61-3
Edit Defaults mode, 61-5
Edit mode, 61-4
Help mode, 61-5
HTML for portlets, 61-2
portlet navigation, 61-6
Preview mode, 61-5
task flow creation, D-1

H

height portlet tag attribute, 64-25
HelpPage.jsp file, A-8
hideActions property, 40-6, 49-13
hideAttach property, 40-6
hideComments property, 32-13, 32-14, 40-6
hideConfigure property, 40-6
hideDocumentUploader property, 40-7
hideFooter property, 40-7
hideGiven property, 40-7
hideHeader property, 40-7
hideInlinePreview property, 40-7
hideLike property, 40-7
hidePublisher property, 40-8
hideRefiners property, 45-11
hideScopePicker property, 40-8
hideToolbar property, 33-10
hierarchy, portal, 9-5
hintsTextKey property, 40-8
History pane, 31-37, 31-39
home page, changing, 9-26
HTML, guidelines for portlets, 61-2
htmlFormHiddenFields utility, 61-47
HTTP methods, REST APIs, 54-11
HTTP response status codes
REST APIs, 54-12
HTTPS, Web Clipping, E-18

I

icon portlet tag attribute, 64-25
icon property, 19-10
Icon View display template, 31-25
id portlet tag attribute, 64-24
identity propagation mechanisms, 69-44
IFRAME, 64-33
iframeDtd portlet tag attribute, 64-32
images
document component, 31-11
enabling Profile photo upload, 38-10, 39-2
resource proxy, 61-8
imageSize property, 40-8
imageURL property, 32-12
IMP (Instant Messaging and Presence) service
about, 35-1
adding to Framework applications, 35-10
connections, 35-3
customizing views, 35-19
enabling network presence, 35-15
Presence component, 35-11
Presence Data component, 35-11
requirements, 35-2
security, 35-12
troubleshooting, 35-20
importing
customizations for WSRP 2.0, 61-34
deployed resources into JDeveloper, 17-16
includeCatalog, C-13
index.html file, 9-26
initialSidebar property, 31-37, 31-39
inline frames, 31-11, 58-2
inlineStyle portlet tag attribute, 64-31
installing
JDeveloper, 1-3
WebCenter Portal Extension Bundle, 1-3
Instant Messaging and Presence (IMP)
See IMP
Integrated WebLogic Server
runtime customizations, preserving, 9-19
See Integrated WLS
See WebLogic Server, 9-19
Integrated WLS
deploying JSR 286 portlets, 62-3
deploying PDK-Java portlets, 62-5
determining if it is running, 3-2
index.html file, 3-6
readme file, 3-3
starting, 3-2
stopping, 3-2
testing Framework applications, 70-2
testing JSR 286 portlets, 62-3
testing PDK-Java portlets, 62-5
integrating
E-Business Suite applications, 55-9
JD Edwards applications, 55-14
Oracle applications, 55-1
Oracle Business Intelligence Presentation Services, 55-24
PeopleSoft applications, 55-17

Siebel applications, 55-2
internal links, 61-8
interportlet communication within JSF portlets, 59-3, 59-6, 59-8
intraportlet links
about, 61-7
URL parameters, 61-43
inviteMember API, 57-17
isAboutModeAvailable portlet tag attribute, 64-29
isCategoryView property, 34-10
isConfigModeAvailable portlet tag attribute, 64-29
isCustomizeModeAvailable portlet tag attribute, 64-29
isDetailModeAvailable portlet tag attribute, 64-29
isDiscoverable API, 57-21
isHelpModeAvailable portlet tag attribute, 64-30
isNormalModeAvailable portlet tag attribute, 64-30
ISO-8859-1 character set, 66-9
isPersonalizeModeAvailable portlet tag attribute, 64-30
isPreviewModeAvailable portlet tag attribute, 64-30
isPrintModeAvailable portlet tag attribute, 64-30
isPublic API, 57-21
isUpdateStatus property, 40-8
items, reusing from Oracle Portal, F-10
iterative development, 9-17
about, 1-4
disabling, 1-6
enabling, 1-6

J

J2EE security roles, mapping to WSRP producer, 64-10
Java APIs
Notifications and, 49-14
People Connections service, 41-10
Personalization, 68-1
Search service, 45-17
Java Community Process, 58-16
Java EE application server, 58-18
Java Object Cache, disabling for preference store, 61-36
Java portlets
about, 58-9
authenticating to external applications, 58-29
authentication security manager, 58-28
caching style, 58-22
capturing content, 58-25
charting, 58-26
creating JSR 286 portlets, 60-5
creating PDK-Java portlets, 60-14
creation style, 58-23
design-time flexibility, 58-24
development tools, 58-22
expertise required, 58-17
group security manager, 58-28
guidelines, 61-1
hiding and showing, 58-28
intended users, 58-9
multilingual support, 58-28
pagination support, 58-28
parameter support, 58-27
rendering inline, 58-26
usage suitability, 58-16
use case, 58-9
JavaScript, Web Clipping, 66-8
JavaServer Faces
See JSF
javax.faces.DEFAULT_SUFFIX parameter, 1-10
jazn-data.xml, H-16
JCR 1.0, 28-1
JCR adapters, 2-4
JD Edwards applications, 55-14
JDBC data source
application-level data source, 70-7
global data source, 70-7
JDeveloper
about, 2-10
installing, 3-1
JNDI variables, 61-51
declaring, 61-52
naming conventions, 61-52
PDK-Java JNDI variables, 61-55
persistentStore variable, E-22
retrieving, 61-54
setting values, 61-53
types, 61-52
JPS portlets files, A-4
about.jsp, A-7
catalog-registry.xml, A-2
config.jsp, A-7
default-catalog.xml, A-3
default-navigation-model.xml, A-3
edit_defaults.jsp, A-7
edit.jsp, A-7
error.jspx, A-3
faces-config.xml, A-13
help.jsp, A-7
home.jspx, A-3
index.html, A-3
JSPX file, A-13
login.jspx, A-3
mds subdirectory, A-15
navigation-registry.xml, A-3
navigation-renderer.jspx, A-4
oracle-portlet-tags.jar, A-6
PageDef.xml, A-13
PageDef.xml, sample file, A-13
pages.xml, A-3
pageTemplate_globe.jspx, A-4
pageTemplate_swooshy.jspx, A-4
portlet_mode.jsp, A-6
portlet_nameBundle.jar, A-7
portlet_name.java, A-7
portlet.xml, A-4
portlet.xml, sample file, A-4
preview.jsp, A-7
print.jsp, A-7
trinidad-config.xml, A-15

view.jsp, A-7
web.xml, A-7, A-15
wsdl subdirectory, A-15
JSF applications, creating portlets, 59-2
JSF portlets
about, 58-4
comparing portlet building technologies, 58-13
creating from JSF applications, 59-2
guidelines
general guidelines, 59-10
JSF guidelines, 59-13
Oracle ADF guidelines, 59-13
portlet guidelines, 59-11
security guidelines, 59-12
intended users, 58-4
portlet entry for task flow, 59-7
portlet events, 59-3, 59-6, 59-8
testing, 59-8
use case, 58-5
JSF servlet settings, 1-9
JSP servlet, 1-10
JSR 286 portlets
about JSR 286, 60-2
adding personalization, 61-26
caching, 61-31 to 61-34
comparing portlet building technologies, 58-13
creating, 60-5
deploying, 62-3
expiry-based caching, 61-32
guidelines, 61-1
JSR 286 and WSRP, 60-2
security, 61-36
testing, 62-3
validation-based caching, 61-33

K

keepOpenAfterPublish property, 40-8
keywordsInputRendered property, 45-11

L

label property, 19-10
lastModifiedAfter property, 31-28, 31-36
lastModifiedBefore property, 31-28, 31-36
lastModifier property, 31-28, 31-36
Layout Customizable
about, 18-49
adding to a page, 20-7
layout types, B-10
properties, B-6
style selectors, B-56
layout property, 31-30
life cycle, portal, 9-4
about, 10-1
adding content, 10-9
building and testing environments, 10-6
deploy and configure scripts
about, 10-12
using, 10-12
life cycle tasks
development tasks, 10-4
nightly build, 10-5
one-time setup tasks, 10-4
production, 10-5
staging, 10-5
testing, 10-5
managed servers, 10-12
managing security, 10-23
moving
portal to existing environment, 10-11
portal to production, 10-9
nightly builds, 10-10
phases of, 10-1
portlet preferences, 10-25
propagating WebCenter Content content, 10-23
propagation tool, 10-18
setting up production environment, first time, 10-11
staging and production environments, 10-8
tools for managing, 10-10
who participates in, 10-6
links
bidirectional links, 43-3
building with URL parameters, 61-45
document component, 31-11
linking from objects, 43-2
linking to existing objects, 43-2
linking to new objects, 43-2
linking with other WebCenter services, 43-2
types of portlet links
application, 61-8
external, 61-8
internal or resource, 61-8
intraportlet, 61-7
Links pane, 31-39
Links service, 43-3
about, 43-1
connection, 43-4
custom JSF components
Links Detail Button, 43-2
Links Detail Menu Item, 43-2
Links Status Icon, 43-2
requirements, 43-3
security, 43-6
troubleshooting, 43-6
List of Resource Types task flow, 17-6
List View display template, 31-25
List with Details Panel View display template, 31-25
Lists service, 48-1 to 48-34
about, 48-1
adding at design time, 48-2
adding to Resource Catalog, 48-4
configuring, 48-2
enabling runtime use of, 48-3
required permissions, 48-1
REST APIs, 48-4 to 48-34
troubleshooting, 48-34
Lists task flow, about, 48-2
local data sources, 70-13

locators, 67-48
Login Metrics task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsmaxDataPointsPerSeries, 47-14
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
login pages
creating, 69-22
setting, 9-29
login portlet, 69-23
logout page, setting, 9-29
look and feel, portal, 9-16
LOVs (list of values), for task flow parameters, 21-22

M

Mail Compose page, 36-14
mail server
about, 36-3
Microsoft Exchange, 36-3
notifications and, 49-5
Mail service
about, 36-1
adding to a Framework application, 36-11
configuring the number of mails displayed, 36-15
connection, 36-5
customizing views, 36-14
distribution lists, 36-3
Mail Compose page, 36-14
Mail task flows
, 36-11
message filters, recent messages, 36-2
requirements, 36-3
security, 36-14
troubleshooting, 36-16
Mail task flow
about, 36-11
tabularView parameter, 36-13
managed server
creating, 70-12
provisioning, 70-12
managed servers, deploying to, 10-12
mashup styles
about, 12-6
creating, 12-26
enabling runtime administration, 12-28
exporting, 12-30
importing, 12-29
in WebCenter Portal: Spaces, 12-30
maxDocuments property, 31-36
maxResults property, 31-23
MDS (Metadata Services), configuring for Composer, 23-1
mds subdirectory, A-15
mds-config element, B-41
Message Board
about, 38-7
REST APIs, 42-26
supported site-level settings, 39-8
task flows
Message Board, 38-7
Message Board - Quick View, 38-8
Message Board - Quick View task flow
about, 38-8
parameters
displayCount, 40-5
hideActions, 40-6
hideFooter, 40-7
hideHeader, 40-7
hidePublisher, 40-8
spaceName, 40-10
userId, 40-11
Message Board task flow
about, 38-7
parameters
displayCount, 40-5
fromDate, 40-6
hideHeader, 40-7
hidePublisher, 40-8
resourceId, 40-10
spaceName, 40-10
toDate, 40-11
messaging channels, 49-1
Metadata Service Repository schema
creating, 70-12
registering, 70-12
migrating applications
about, H-1, H-2
configuring security, H-16
migrating PS2 WebCenter custom applications, H-1
preparing, H-2
mimeType property, 45-12
Mini Properties task flow
See Document Mini Properties task flow
mod_osso, Web Clipping, 66-8
modes
See portlet modes
month property, 32-13
mostRecentFirst property, 31-36
multilingual support, portlets, 58-28

N

name portable contact type, 54-15
namespace, Notifications and, 49-4
navigateToAnnouncementViewer property, 33-10
navigation
creating navigation models, 11-2
EL expressions
about, 14-10
accessing current navigation model, 14-11
accessing default navigation model, 14-11
accessing named navigation model, 14-11
accessing node properties, 14-14
accessing specific nodes, 14-13
binding model to UI components, 14-12

Navigation Context, 14-11
rendering as a list of links (example), 14-16
rendering as a menu (example), 14-17
rendering as a tree (example), 14-17
rendering as breadcrumbs (example), 14-18
rendering master-detail (example), 14-19
setting the runtime model, 14-12
EL expressions relating to, G-6
link API, 61-45
linking portlets, 61-48
model, 9-11
navigating within a Java portlet, 61-6
navigation content, 11-1
navigation models, 11-1
navigation structure, 11-1
registry, 9-11
task flows
adding to a page template, 14-3
Navigation - Breadcrumbs, 14-2
Navigation - Menu, 14-2
Navigation - Tree, 14-2
style classes, 14-9
visualizing, UI
about, 14-1
using ELs, 14-10
using REST APIs, 14-21
Navigation - Breadcrumb task flow
about, 14-2
associating navigation model, 14-9
parameters, 14-9
navigationFile, 14-9
showRoot, 14-9
startNode, 14-9
style, 14-9
Navigation - Menu task flow
about, 14-2
associating navigation model, 14-7
parameters, 14-7
navigationFile, 14-7
showIcons, 14-8
startNode, 14-7
style, 14-8
subStyle, 14-8
Navigation - Tree task flow
about, 14-2
associating navigation model, 14-5
parameters
depth, 14-6
navigationFile, 14-5
showIcons, 14-6
showRoot, 14-5
startNode, 14-5
style, 14-6
navigation bar, portal, 9-9
navigation models
about, 9-3, 11-1
adding resources
content queries, 11-7
custom components, 11-18
custom content providers, 11-21
custom folders, 11-20
custom resources, 11-17
direct links, 11-24
documents, 11-5
external applications, 11-9
page hierarchy, 11-13
pages, 11-9
portlets, 11-9
task flows, 11-9
URLs, 11-9
advanced content query, 11-8
creating, 11-2
default model, 11-2
delegated maintenance, 11-15
display options, 11-23
exporting, 11-29
importing, 11-29
in WebCenter Portal: Spaces, 11-29
navigation registry, 11-28
navigation renderer, 11-26
nesting, 11-15
organizing, 11-23
runtime management, 11-27
selecting default, 11-25
using Content Presenter templates, 11-7, 11-9
navigation registries, 9-3, 11-28
navigation renderer, 11-26
navigation visualization
about, 14-1
using ELs, 14-10
using REST APIs, 14-21
navigational parameters
See public render parameters
navigationFile property, 14-5, 14-7, 14-9
navigation-registry.xml file, 11-28
node classes (Activity Graph)
about, 46-2
defaults for WebCenter Portal, 46-2
extending, 46-25
Notifications, 49-1 to 49-20
about, 49-1
activity triggers, 49-1
adding namespace, 49-4
at runtime, 49-3
configuration file, 49-2
configuring, 49-4 to 49-14
connecting to BPEL server, 49-6
connecting to mail server, 49-5
delivery platforms, 49-2
effects of configurations
on Connections, 49-10
on Feedback, 49-11
on Message Board, 49-10
Java APIs, 49-14
messaging channels, 49-1
participant services and objects, 49-1
requirements, 49-2
setting application defaults, 49-7
setting security, 49-13
task flows

adding to a page, 49-12
input parameters, 49-13
Subscription Preferences, 49-3, 49-11
Subscription Viewer, 49-3, 49-11
using data controls, 49-16
numberOfRows_list property, 40-8

O

objectId property, 40-8
objectName property, 46-16
objectType property, 40-8
OCS Global Profile, node properties, 29-25
OmniPortlet, 3-4
about, 58-8, 65-1
adding to a page, 65-3
AllModesSharedScreen property, 64-20
authenticating to external applications, 58-29
caching style, 58-22
capturing content, 58-25
charting, 58-26
comparing portlet building technologies, 58-13
configuring to access data outside a firewall, E-2
configuring to access relational databases, E-5
creation style, 58-23
customizing, 65-3
data sources, 58-17
design-time flexibility, 58-24
development tools, 58-22
expertise required, 58-17
hiding and showing, 58-28
installing DataDirect JDBC drivers, E-5
intended users, 58-9
multilingual support, 58-28
pagination support, 58-28
parameter support, 58-27
provider.xml file, E-5
registering DataDirect JDBC drivers, E-5
registering the producer, 65-2
rendering inline, 58-26
troubleshooting, 65-4
usage suitability, 58-16
use case, 58-9
OpenUsage API
about, 47-2
enabling, 47-7
Oracle ADF (Oracle Application Development Framework)
See ADF
Oracle Business Intelligence Presentation Services, 55-24
Oracle Discoverer, charting, 58-26
Oracle JSF Portlet Bridge, 59-1
authenticating to external applications, 58-28
caching style, 58-22
capturing content, 58-24
charting, 58-26
design-time flexibility, 58-24
development tool, 58-22
expertise required, 58-17
general suitability, 58-15
guidelines
general guidelines, 59-10
Oracle ADF guidelines, 59-13
portlet guidelines, 59-11
security guidelines, 59-12
hiding and showing, 58-27
JSF portlets, 58-4
pagination support, 58-28
parameter support, 58-27
portlet creation style, 58-23
rendering inline, 58-25
runtime-created skins, 59-16
testing portlets, 59-8
undeclared events, 59-15
See also JSF portlets
Oracle PDK-Java portlets
See PDK-Java portlets
Oracle Portal
configuration parameters, 27-8
FPA (Federated Portal Adapter), F-4
reusing components in WebCenter Portal, F-1
reusing items, F-10
reusing portlets, F-2
Web Clipping limitations, 66-8
Oracle Reports, charting, 58-26
Oracle SES (Oracle Secure Enterprise Search)
connection, 45-6
using with WebCenter, 45-15
Oracle WebCenter Portal services
See WebCenter Portal services
Oracle WebCenter Portal: Framework
JSR 286, 58-13
portlet technologies, 58-12
Struts integration, 61-62
WSRP, 58-13
WSRP support, 60-3
Oracle WebLogic Server
Integrated WLS Server, 70-1
managed servers, 70-2
organization portable contact type, 54-16
orgBreadcrumbsShown property, 40-9
outcome property, 19-9
overview
security, 69-2
WSRP producers security, 69-45

P

Page - Create New task flow
about, 19-2
adding, 19-3
parameters
accessing, 19-8
adftmplate, 19-9
icon, 19-10
label, 19-10
outcome, 19-9
scopename, 19-9
templatefile, 19-9

uitype, 19-10
Page Composer
See Composer
Page Customizable
about, 18-47
adding to a page, 20-3
properties, B-2
style selectors, B-54
page hierarchy, 9-5, 69-17
about, 9-3
building a page hierarchy, 69-20
delegating security, 69-21
editing, 9-7
page hierarchy model, 69-17
Page Hierarchy Security Editor, 69-17
Page service
about, 19-1
adding to applications, 19-3
APIs, 19-1, 19-18, 19-33
data control, 19-1, 19-18
Page - Create New task flow, 19-2
samples of Page service functionality, 19-42
scopes, 19-1
security, 19-4
page styles
about, 12-4
creating, 12-19
enabling runtime administration, 12-22
exporting, 12-24
importing, 12-23
in WebCenter Portal: Spaces, 12-24
page templates, 9-5
about, 9-4, 12-1, 12-2, 12-7
adding navigation UI, 12-13
adding navigation UI using ELs, 14-10
ADFm page definition, 12-14
applying to pages at design time, 12-2
applying to pages at runtime, 12-3
attributes, 12-11
content facet, 12-11, 12-13, 12-14
Content Presenter task flow, 12-15
creating, 12-9, 19-19
default page templates, 12-8
deleting, 12-17
editing, 12-17
enabling runtime management, 12-3, 12-17
enabling runtime switching, 12-15
in WebCenter Portal applications, 12-14
in WebCenter Spaces, 12-19
mailto links, 12-14
Page Traffic task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
page variables, deleting for portlets, 64-38
PAGE_LINK URL, 61-44
page-level security, 69-39
pages, 9-5
adding
files, 31-12
folders, 31-12
adding hierarchy to portal navigation, 11-13
adding portlets, 64-17
adding to portal navigation, 11-9
changing home page, 9-26
consuming portlets, 64-1
creating, 5-8, 19-1
creating portlets, 59-2
Edit mode, 18-3
managing, 19-18
portlet entry, 59-3
requirements for WebCenter Portal applications, 5-8
schemes
See page templates
styles
See page styles
templates
creating, 19-19
View mode, 18-3
pages files, A-10
adf-config.xml, A-10
adf-portlet-config sample, A-12
DataBindings.cpx, A-13
pageSize property, 32-13, 32-14, 33-10, 40-9
Document Explorer task flow, 31-26
Document List Viewer task flow, 31-29
Document Manager task flow, 31-31
Document Navigator task flow, 31-32
Folder Viewer task flow, 31-34
pagination property, 40-9
pagination support, portlets, 58-28
Panel Customizable
about, 18-48
adding to a page, 20-6
applying action-level restrictions, 24-16
disabling Edit option, B-15
properties, B-10
style selectors, B-56
Parameter Form portlet
about, 58-6
intended users, 58-6
use case, 58-7
parameters
blog task flows, 32-12
building links with URL types, 61-45
context parameters, 1-10
javax.faces.DEFAULT_SUFFIX, 1-10
PDK-Java portlets
private parameters, 61-43
public parameters, 61-38
portlets, 58-26
private portlet parameters, 58-27
public portlet parameters, 58-26
URL, 61-43
See also properties

parentId property, 33-10
partialTriggers portlet tag attribute, 64-25, 64-29
partner portlets, 58-5
PDK-Java portlet files, A-7
_default.properties, A-8
connections.xml, A-16
index.jsp, A-8
portlet_nameAboutPage.jsp, A-8
portlet_nameEditDefaultsPage.jsp, A-8
portlet_nameEditPage.jsp, A-8
portlet_nameHelpPage.jsp, A-8
portlet_nameShowDetailsPage.jsp, A-8
portlet_nameShowPage.jsp, A-8
producer_name.properties, A-7
provider.xml, A-8
provider.xml sample, A-9
provider.xml syntax, A-9
web.xml, A-9
PDK-Java portlets
about PDK-Java, 60-3
adding portlet modes, 61-37
authentication, 69-49
building links with URL types, 61-45
caching, 61-58 to 61-61
communication, 69-49
comparing portlet building technologies, 58-13
creating, 60-14
creating struts portlets, 61-61
escaping user inputs, 69-53
expiry-based caching, 61-60
external application, 69-50
guidelines, 61-1
JNDI variables, 61-55
message authentication, 69-50, 69-52
message encryption, 69-50
no authentication, 69-51
Preference Store Migration/Upgrade Utility, E-27
private parameters, 61-43
public parameters, 61-38
security, 69-49
security managers, 69-51
server authentication, 69-50
session storage, 61-55 to 61-58
checking for valid session, 61-57
enabling in provider.xml, 61-58
implementing, 61-56
single sign-on, 69-50
Struts integration, 61-62
validation-based caching, 61-61
PDK-Java producers, 3-6, 58-18
deleting portlets, 64-38
deregistering, 64-16
editing registration values, 64-15
enabling producer sessions, 64-14
external applications, 64-13
refreshing, 64-16
registering, 64-11
service ID, 64-13
syntax of URL endpoint, 64-12
testing connection, 64-16
pe_ext.xml, 21-4, B-29
People Connections
People Connections locator, 67-6
Personalization, 67-6
People Connections service, 38-1 to 38-12
about, 38-1 to 38-12
Activity Stream, 38-2, 38-3
Advanced Query option, 40-12
REST APIs, 42-1
supported site-level settings, 39-5
configuring, 39-1
Connections, 38-5
accepting invitations automatically, 39-11
example site-level settings, 39-11
REST APIs, 42-10
supported site-level settings, 39-6
data controls, 41-1 to 41-10
People Connections Management data control, 41-3
Profile data control, 41-10
database set-up, 39-1
EL expressions relating to, G-14
Feedback
about, 38-6
REST APIs, 42-23
supported site-level settings, 39-7
Java APIs, 41-10
Message Board
about, 38-7
REST APIs, 42-26
supported site-level settings, 39-8
Profile
about, 38-9
field names, 39-10
REST APIs, 42-10
section names, 39-10
supported site-level settings, 39-9
requirements, 38-12
REST APIs, 42-1 to 42-30, 54-24, 54-27
security, 39-2
site-level settings
preparing and running settings-insert.sql, 39-3
preparing and running settings-update.sql, 39-4
scripts, 39-3
supported settings, 39-5
task flow binding parameters
about, 40-1
parameters table, 40-3
revising, 40-1
task flows
Activity Stream, 38-4
Activity Stream - Quick View, 38-4
Connections, 38-5
Connections - Card, 38-5
Connections - Quick View, 38-6
Feedback, 38-6
Feedback - Quick View, 38-7
Message Board, 38-7

Message Board - Quick View, 38-8
Profile, 38-9
Profile - Snapshot, 38-9
Profile Gallery, 38-10
Publisher, 38-11
PeopleSoft applications, 55-17
persist-changes attribute, 24-15
persistence change manager
about, 22-27
configuring, 22-29
Personalization
about, 67-1
Activity Graph provider, 67-5, 67-19
adding custom locators, 67-48
adding custom providers, 67-39
architecture, 67-2
calling data providers, 67-14
calling function providers, 67-13
CMIS Provider, 67-5
CMIS provider, 67-16
Conductor, 67-2
Conductor REST APIs, 68-96
creating a new scenario, 67-10
custom data providers, 67-6
data providers, 67-4, 67-14
Expression Language APIs, 67-3, 68-195
extending, 67-38
file-based scenarios, 67-23
function providers, 67-4, 67-6, 67-13
Java APIs, 68-1
People Connections locator, 67-6
presenting content, 67-27
Properties Namespace file, 67-8
property definitions, 67-9
Property Editor, 67-13
property locators, 67-4
Property Service, 67-3
Property Service provider, 67-16
Property Service REST APIs, 68-1, 68-24
property sets, 67-3, 67-4, 67-9
providers, 67-3, 67-4
requirements, 67-7
Scenario Editor, 67-3
scenario node types, 67-12
scenarios, 67-2
personalization
adding to JSR 286 portlets, 61-26
building and deploying, 10-15
deploy to archive file, 10-17
MAR file, building, 10-15
propagation of files, 10-22
testing for portlets, 61-61
Personalizatoin in WebCenter Portal
See Personalization, 67-1, 68-1
personReference data type, 54-14
photoUploadAllowed binding parameter, 38-10, 39-2
photoUploadAllowed property, 40-9
planning WebCenter Portal applications
about, 2-1
design considerations, 2-12
developer considerations, 2-14
site administrator considerations, 2-14
user considerations, 2-12
pollId property, 37-5
Polls - Polls Manager task flow
about, 37-3
parameters
scope, 37-5
showUserDataOnly, 37-5
Polls - Quick Poll task flow
about, 37-3
showInEditMode parameter, 37-5
Polls - Take Polls task flow
about, 37-3
pollId parameter, 37-5
Polls - View Poll Results task flow
about, 37-3
pollId parameter, 37-5
Polls service
about, 37-1
adding to Framework applications, 37-3
connection, 37-3
requirements, 37-2
task flows
Polls Manager, 37-3
Quick Poll task flow, 37-3
Take Polls, 37-3
View Poll Results, 37-3
portable contact types
address, 54-15
name, 54-15
organization, 54-16
value, 54-16
portal
about, 9-1
administration console, 9-20
at runtime, 9-19
catalog registry, 9-13
catalogs, 9-4
registry, 9-4
creating, 9-4
customization, 9-4
default home page, changing, 9-26
delegated administration, 9-4
designing, 9-5
development, 9-15
features, 9-2
files, organization of, 9-22
framework, 9-3
index page, 9-28
iterative development, 9-17
life cycle, 9-4
login page, 9-29
logout page, 9-29
look and feel, 9-16
mashup styles, 12-25
navigation model, 9-3, 9-11
navigation registry, 9-3, 9-11
navigation, *See* portal navigation, 9-15

organization in JDeveloper, 9-25
page hierarchies, 9-3
page styles, 12-19
page templates, 9-5
pages, 9-5
preferences, 9-4, 9-16
preferences, changing at runtime, 9-17
project on filesystem, 9-24
resource catalog, 9-13, 16-1
resources, runtime editing, 9-21
round-trip development, 9-17
running, 9-18
securing pages, 9-9
security at runtime, 9-21
skins, 9-4, 9-16
sources folder, 9-18
testing, 9-18
portal application
See Framework applications
portal navigation
navigation models, 11-1
navigation registry, 11-28
navigation renderer, 11-26
navigation UI, 14-1
navigation visualization, 14-1
partial page refresh, 11-27
portal-skin.css
about, 15-4
editing, 15-5
portlet applications
creating, 5-12
Portlet Producer Application template, 5-13
portlet bridge
See Oracle JSF Portlet Bridge
Portlet Consumer Error, 64-59
Portlet Consumer Test Page, 64-50
portlet events
adding processing events, 61-22
adding publishing events, 61-23
setting at application level, 61-21
portlet filters
about, 61-29
adding to an application, 61-29
applying to portlets, 61-30
Portlet Instance Response Time task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
Portlet Instance Traffic task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
portlet Java class, A-7
portlet modes
about, 61-1
adding for PDK-Java portlets, 61-37
portlet preference store, E-21
Portlet Producer Application template, 5-13
portlet producers
applications, creating, 5-12
architecture, 58-20
character set for PDK-Java, 60-17
combining portlets from different portlet producers, 4-13
connections, 1-7, 4-13
considerations, 4-13
deleting portlets, 64-38
deregistering, 64-16
editing registration values, 64-15
exporting at design time, E-31
files affected, 4-6
importing at design time, E-32
Integrated WLS and, 3-4
name clashes, 4-13
PDK-Java, 3-6
PDK-Java producers, 58-18
PDK-Java, registering, 64-11
portlet deployment, 58-18
preconfigured, 3-2
properties file, A-7
refreshing, 64-16
registering, 64-2
for PDK-Java, 62-13
for WSRP, 62-13
OmniPortlet, 65-2
registering with WebCenter Portal: Spaces, 57-3
request flow, 58-20
testing connection, 64-16
WSRP producers, 58-19
WSRP, registering, 64-3
See also PDK-Java producers
See also WSRP producers
Portlet Response Time task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
portlet security managers, 69-51
portlet tag, placement, 64-20
Portlet Timeout error, 64-59
Portlet Traffic task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
portlets

about, 2-3, 58-1
accessing user information, 61-11
Actions menu, 58-3
adding
  personalization to JSR 286 portlets, 61-26
  portlet modes for JSR 286 portlets, 61-10
  portlet modes for PDK-Java portlets, 61-37
  processing events, 61-22
  public render parameters, 61-25
  publishing events, 61-23
  to pages, 64-17
  to portal navigation, 11-9
adding to WebCenter Portal: Spaces, 57-3
adfp:portlet tag
  See adfp:portlet tag
attributes, 64-21
authentication, 64-21
automatic wiring, 64-20
binding attributes, 64-31
caching, 58-21
capturing content, 58-24
character set for PDK-Java, 60-17
charting, 58-14, 58-26
choosing a tool, 58-10
chrome, 58-3
comparing portlet building technologies, 58-13
consuming, 64-1
copying
  about, 64-34
  to another page, 64-36
  to the same page, 64-34
creating
  JSF portlets from pages, 59-2
  JSF portlets from task flows, 59-5
  JSR 286 portlets, 60-5
  PDK-Java portlets, 60-14
  Struts portlets, 61-61
creation style, 58-22
customization, 58-3
declarative development, 58-10
deleting, 64-38
deploying, 58-18, 62-1
deployment descriptor, 62-10
deregistering portlet producers, 64-16
design time at runtime, 3-4, 58-14
development tools, 58-22
editing portlet.xml, 61-9
errors
  Portlet Consumer Error, 64-59
  Portlet Timeout, 64-59
  Remote Portlet Communication Error, 64-61
  Remote Portlet Error, 64-62
files affected, 4-7
guidelines
  About mode, 61-5
  CSS, 61-3
  Edit Defaults mode, 61-5
  Edit mode, 61-4
  Help mode, 61-5
  HTML, 61-2
  Java, 61-1
  navigation, 61-6
  Preview mode, 61-5
hiding and showing, 58-27
IFRAME, 64-33
implementing internal navigation, 61-48
in Framework applications, 64-1
inline content rendering, 58-14
inline frames, 58-2
Integrated WLS and, 3-4
Java portlets, about, 58-9
JSF portlets, 59-1
JSF portlets, about, 58-4
JSP files for PDK-Java, A-8
links
  applications, 61-8
  external, 61-8
  internal, 61-8
  intraportlet, 61-7
  resource, 61-8
logging file, 64-57
metric task flows, 47-2
modes
  See portlet modes
multilingual support, 58-28
navigation link API, 61-45
OmniPortlet, 3-4, 58-8
Oracle JSF Portlet Bridge, 59-1
overriding inherited security, 69-41
pagination support, 58-28
Parameter Form portlet, 58-6
PDK-Java portlets, 69-49
personalization, 58-3, 61-61
Portlet Consumer Test Page, 64-50
portlet entry for page, 59-3
portlet events, 61-21
portlet filters, 61-29
portlet parameters
  private, 58-14, 58-27, 61-43
  public, 58-14, 58-26, 61-38
portlet producers, 58-18
portlet tag
  appearance attributes, 64-25
  behavior attributes, 64-28
  common attributes, 64-23
  customization attributes, 64-31
  other attributes, 64-31
  portlet mode attributes, 64-29
  style attributes, 64-30
  See also adfp:portlet tag
prebuilt, 3-2
prebuilt portlets, 58-5
preferences, migration of, 10-25
Producer Test Page, 64-54
programmatic development, 58-10
refreshing portlet producers, 64-16
registering portlet producers, 64-2
rendering inline, 58-25
resource bundle, A-7
reusing from Oracle Portal, F-2

running, 64-20
sample portlets, 3-5, 3-6
security, 58-3, 58-14, 69-41
security at actions category level, 69-41
security at actions level, 69-42
skin style selectors, B-58
testing, 62-1
troubleshooting, 64-48
URL types, 61-44
WAR deployment profile, 62-8
Web Clipping portlet, 3-4, 58-7
web content capture, 58-14
portlet.xml
adding portlet filters, 61-29
adding processing events to portlets, 61-22
adding public render parameters to portlets, 61-25
adding publishing events to portlets, 61-23
applying portlet filters, 61-30
container runtime options, 61-12
defining public render parameters at application level, 61-24
editing, 61-9
setting CROs at application level, 61-19
setting events at application level, 61-21
prebuilt portlets, 58-5
preference store
consumer (WSRP producer), E-22
database (WSRP producer), E-22
disabling Java Object Cache, 61-36
file-based (WSRP producer), E-22
migration utility, E-24
WSRP container, E-24
Preference Store Migration/Upgrade Utility, E-24
preferences
changing at runtime, 9-17
portal, 9-4, 9-16
preferenceStore tag
class attribute, E-23
connection parameter, E-23
name parameter, E-23
Presence component, 35-11
presence servers
LCS, 35-2
Lync, 35-2
OCS, 35-2
Preview mode, guidelines, 61-5
printStackTrace API, 57-27
private parameters
in PDK-Java portlets, 61-43
restricting navigation, 61-51
Producer task flow, 64-47
Producer Test Page, 64-54
producers
See portlet producers
production environment, 10-8
production environment tasks, 10-5
Profile
about, 38-9
EL expressions relating to, G-14
enabling photo upload, 38-10, 39-2
field names, 39-10
REST APIs, 42-10
section names, 39-10
supported site-level settings, 39-9
task flows
Profile, 38-9
Profile - Snapshot, 38-9
Profile Gallery, 38-10
Profile - Snapshot task flow
about, 38-9
parameters
imageSize, 40-8
orgBreadCrumbsShown, 40-9
photoUploadAllowed, 40-9
profileEditLinkShown, 40-9
statusUpdateAllowed, 40-11
userId, 40-11
Profile Gallery task flow
about, 38-10
resourceId parameter, 40-10
Profile task flow
about, 38-9
parameters
editMode, 40-5
resourceId, 40-10
sectionWiseEditEnabled, 40-10
profileEditLinkShown property, 40-9
profileLaunchedOnSelection property, 40-9
profileOnly property, 40-9
profileUserId property, 40-9
projection parameter (REST APIs), 54-10
propagation
of WCPS files, 10-22
of WebCenter Content content, 10-23
tool, 10-18
properties
_followEnforced, 40-3
_objectExtHandlerEnabled, 40-3
_objects, 40-3
_objectSecured, 40-3
adftemplate, 19-9
advancedQuery, 40-3
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
categoryId, 34-10
classURN, 46-16
classURNRestrictions, 46-17
connectionListName, 40-4
connectionName
Document Explorer task flow, 31-26
Document List Viewer task flow, 31-28
Document Manager task flow, 31-30
Document Navigator task flow, 31-32
Folder Viewer task flow, 31-33
Recent Documents task flow, 31-35
content items, 29-25
content type, 29-15
createdAfter, 31-28

createdBefore, 31-28
creator, 31-28
currentView, 40-4
customAttributes, 45-11
datasource, 31-22
datasourceType, 31-23
defaultTopicFetchSize, 34-10
depth, 14-6
destinationID, 31-39, 31-40
destinationSelectionEnabled, 31-40
disableToolbar, 34-10
display_maxConnections, 40-4
display_numberOfColumns, 40-4
display_numberOfRows, 40-4
display_profileFormat, 40-5
display_removalAllowed, 40-5
display_sortedBy, 40-5
displayCount, 40-5
displayMessageSize, 40-5
doNotAllowSelectingPageSize, 34-10
doNotShowMoreLink, 34-10
editMode, 40-5
enableContextInfo, 40-6
excludeObjectActions, 46-17
expandAllAnnouncements, 33-10
expandedAnnouncements, 33-10
extAppId, 53-5
featuresOff
  Document Explorer task flow, 31-26
  Document Manager task flow, 31-30
  Document Navigator task flow, 31-32
  Document Upload task flow, 31-40
  Document Viewer task flow, 31-37
  Folder Viewer task flow, 31-34
  Rich Text Editor task flow, 31-39
filterPattern, 40-6
forumId, 34-10
freeFlowView, 33-10
fromDate, 40-6
groupSpace, 31-36
hideActions, 40-6, 49-13
hideAttach, 40-6
hideComments, 32-13, 32-14, 40-6
hideConfigure, 40-6
hideDocumentUploader, 40-7
hideFooter, 40-7
hideGiven, 40-7
hideHeader, 40-7
hideInlinePreview, 40-7
hideLike, 40-7
hidePublisher, 40-8
hideRefiners, 45-11
hideScopePicker, 40-8
hideToolbar, 33-10
hintsTextKey, 40-8
icon, 19-10
imageSize, 40-8
imageURL, 32-12
initialSidebar, 31-37, 31-39
isCategoryView, 34-10
isUpdateStatus, 40-8
keepOpenAfterPublish, 40-8
keywordsInputRendered, 45-11
label, 19-10
lastModifiedAfter, 31-28, 31-36
lastModifiedBefore, 31-28, 31-36
lastModifier, 31-28, 31-36
layout, 31-30
maxDataPointsPerSeries, 47-14
maxDocuments, 31-36
maxResults, 31-23
mimeType, 45-12
month, 32-13
mostRecentFirst, 31-36
navigateToAnnouncementViewer, 33-10
navigationFile, 14-5, 14-7, 14-9
numberOfRows_list, 40-8
objectId, 40-8
objectName, 46-16
objectType, 40-8
orgBreadCrumbsShown, 40-9
outcome, 19-9
pageSize, 32-13, 32-14, 33-10, 40-9
  Document Explorer task flow, 31-26
  Document List Viewer task flow, 31-29
  Document Manager task flow, 31-31
  Document Navigator task flow, 31-32
  Folder Viewer task flow, 31-34
pagination, 40-9
parentId, 33-10
photoUploadAllowed, 40-9
pollId, 37-5
profileEditLinkShown, 40-9
profileLaunchedOnSelection, 40-9
profileOnly, 40-9
profileUserId, 40-9
readOnly
  Document Explorer task flow, 31-26
  Document Manager task flow, 31-31
  Document Navigator task flow, 31-32
  Document Properties task flow, 31-38
  Document Version History task flow, 31-41
  Document Viewer task flow, 31-37
  Folder Viewer task flow, 31-34
recentPostListSize, 32-13
recipe, 46-17
redirectURL, 44-8
regionTemplate, 31-23
resourceID, 32-12, 32-13, 32-14
resourceId, 40-10, 44-8, 45-12, 46-16
  AutoVue task flow, 31-41
  Document Explorer task flow, 31-26
  Document Manager task flow, 31-31
  Document Mini Properties task flow, 31-38
  Document Navigator task flow, 31-32
  Document Properties task flow, 31-38
  Document Version History task flow, 31-41
  Document Viewer task flow, 31-37
  Folder Viewer task flow, 31-34
resourceScope, 44-8, 45-12

resourceType, 45-13, 46-17
rssFeedLocation, 53-5
scope, 37-5, 45-13
scopeId, 40-10
scopeName, 49-13
scopename, 19-9
searchBoxSize, 45-12
searchScope, 45-12
sectionWiseEditEnabled, 40-10
serviceCategories, 40-10
serviceId, 40-10, 44-8, 46-17
serviceIds, 45-12
showDocuments, 31-27, 31-32
showFolders, 31-27, 31-29, 31-34
showIcons, 14-6, 14-8
showInEditMode, 37-5
showRecursiveForums, 34-10
showRoot, 14-5, 14-9
showUserDataOnly, 37-5
spaceName, 40-10
spaces, 40-11
startFolderPath
Document Explorer task flow, 31-27
Document List Viewer task flow, 31-29
Document Manager task flow, 31-31
Document Navigator task flow, 31-33
Folder Viewer task flow, 31-35
startNode, 14-5, 14-7, 14-9
statusUpdateAllowed, 40-11
style, 14-6, 14-8, 14-9
subStyle, 14-8
suppressPopup, 46-17
tabularView, 36-13
taskFlowInstId, 31-24, 31-29
templateCategory, 31-24
templatefile, 19-9
templateView, 31-24
title, 32-12
toDate, 40-11
treeNavCollapsed, 31-27
truncateAt, 33-10
uitype, 19-10
uploadDocumentOnly, 40-11
userId, 40-11
userName, 40-11
viaUser, 40-11
visibleAnnouncements, 33-10
visibleForumsFetchSize, 34-10
visibleTopicsFetchSize, 34-10
year, 32-13
Properties Namespace file, 67-8
Properties pane, 31-37, 31-39
property definitions, 67-9
Property Editor, 67-13
property filters
about, 21-3
defining, 21-48
removing, 21-49
property locators, 67-4
Property Service, 67-3
REST APIs, 68-1, 68-24
Property Service provider, 67-16
property sets, 67-3, 67-4, 67-9
property-panel, B-31
property-panels, B-31
providers, 67-4
See portlet producers
provider.xml file
OmniPortlet, E-5
session storage, 61-58
Web Clipping, E-11
provisioning, staging environment, 10-9
proxy
resource, 61-8
resource proxy for WSRP, 61-35
stateless resource proxying, 61-35
Web Clipping, E-17
proxy server, for REST APIs, 54-16
public parameters, in PDK-Java portlets, 61-38
public render parameters
about, 61-23
adding to portlets, 61-25
defining at application level, 61-24
Publisher task flow
about, 38-11
parameters
hideAttach, 40-6
hideDocumentUploader, 40-7
hideScopePicker, 40-8
hintsTextKey, 40-8
isUpdateStatus, 40-8
keepOpenAfterPublish, 40-8
objectId, 40-8
objectType, 40-8
scopeId, 40-10
serviceId, 40-10
spaceName, 40-10
uploadDocumentOnly, 40-11
userName, 40-11
viaUser, 40-11

Q

q parameter (REST APIs), 54-10
QRPPs (Activity Graph)
about, 46-7
defaults for WebCenter Portal, 46-7
registering, 46-32

R

rank calculations (Activity Graph), extending, 46-29
RCU, 3-7
rcv-config, B-39
readOnly property
Document Explorer task flow, 31-26
Document Manager task flow, 31-31
Document Navigator task flow, 31-32
Document Properties task flow, 31-38
Document Version History task flow, 31-41

Document Viewer task flow, 31-37
Folder Viewer task flow, 31-34
Recent Activities service
about, 50-2
adding, 50-3
compared to Activity Stream, 38-3
connections, 50-3
editing task flow parameters, 50-7
Recent Activities task flow, 50-3
security, 50-7
Recent Blog Posts task flow, 32-6
Recent Documents task flow
about, 31-3
parameters, 31-35
connectionName, 31-35
groupSpace, 31-36
lastModifiedAfter, 31-36
lastModifiedBefore, 31-36
lastModifier, 31-36
maxDocuments, 31-36
mostRecentFirst, 31-36
Recent Posts task flow
parameters
recentPostListSize, 32-13
resourceID, 32-13
Recent Topics task flow
parameters
doNotShowMoreLink, 34-10
visibleTopicsFetchSize, 34-10
recentPostListSize property, 32-13
recipe property, 46-17
recipes
Recommended Connections task flow, 46-12
Similar Items task flow, 46-13
Similar Spaces task flow, 46-12
Recommendations pane, 31-39
Recommended Connections task flow
about, 46-7, 46-11
adding to a page, 46-14
default recipe, 46-12
editing properties, 46-15
parameters
excludeObjectActions, 46-17
recipe, 46-17
resourceId, 46-16
REDIRECT_OPTIONS context parameter, 11-27
redirectURL property, 44-8
regionTemplate property, 31-23
registering
JSR 286 portlet, 62-13
PDK-Java portlet, 62-13
portlet producers, 64-2
resource viewers, 7-14, 44-1
rel attribute, 54-8
Related Items pane, 31-39
RelationshipPermission permission, 43-6
Remote Portlet Communication Error, 64-61
Remote Portlet Error, 64-62
removeMember API, 57-17
rendered portlet tag attribute, 64-27
renderPortletInIFrame portlet tag attribute, 64-26
renderProperty tag, 29-5
Repository Creation Utility (RCU), 3-7
repositoryInfo tag, E-15
class, E-16
connectionName, E-16
databaseHost, E-16
databasePassword, E-17
databasePort, E-16
databaseSid, E-17
databaseUsername, E-17
mdsConfigLocation, E-16
useASO, E-16
useRAA, E-16
reset password page, 69-26
Resource Action Handling framework, 44-20
about, 7-13
custom components, 7-13
resource viewers, 7-14
resource bundle for JPS portlets, portlet_nameBundle.jar, A-7
Resource Catalog
about, 2-5, 16-1
adding lists, 48-4
configuring multiple catalogs, 16-26
configuring the default, 16-20
controlling portlet visibility, 16-18
creating a custom catalog, 16-4
deleting resources, 16-15
editing resources, 16-15
enabling runtime administration, 16-22
example for configuring multiple catalogs, 16-26
example for filtering items, 16-25
exporting, 16-23
exposing design-time data controls, 16-16
exposing runtime data controls, 16-18
factory classes, C-21
filtering items, 16-24
importing, 16-23
location, C-1
managing content, 16-6
navigation registry, 11-28
rearranging resources, 16-15
ResourceCatalogSelector, 16-26
Resource Index, 54-5
resource links, 61-8
Resource Manager
about, 17-1
accessing, 17-8
adding resources, 17-3
adding to a page, 17-6
copying resources, 17-10
creating resources, 17-8
deleting resources, 17-11
downloading resources, 17-16
editing resource details, 17-5
editing resource source code, 17-12
editing resources, 17-10
hiding resources, 17-11
overriding application-level security for

resources, 17-4
previewing resources, 17-13
removing resources, 17-6
setting resource properties, 17-13
setting resource security, 17-11
showing resources, 17-11
uploading resources, 17-19
using at runtime, 17-7
resource proxy, 61-8
for WSRP, 61-35
stateless resource proxying, 61-35
resource string editor
about, 22-31
configuring, 22-30
resource viewer, 7-14, 44-20
ResourceCatalogSelector API, 16-26
resourceID property, 32-12, 32-13, 32-14
resourceId property, 40-10, 44-8, 45-12, 46-16
AutoVue task flow, 31-41
Document Explorer task flow, 31-26
Document Manager task flow, 31-31
Document Mini Properties task flow, 31-38
Document Navigator task flow, 31-32
Document Properties task flow, 31-38
Document Version History task flow, 31-41
Document Viewer task flow, 31-37
Folder Viewer task flow, 31-34
resources
developing for WebCenter Spaces, 56-12
EL expressions relating to, G-3
managing
mashup styles, 12-25
navigation, 14-1
page styles, 12-19
page templates, 12-7
pages, 19-18
Resource Catalogs, 16-1
skins, 15-1
portal, editing at runtime, 9-21
resourceScope property, 44-8, 45-12
resourceType attribute, 54-8
resourceType property, 45-13, 46-17
REST APIs, 68-1, 68-24, 68-96
about, 54-1
Activity Graph service, 46-19
cache management, 54-16
capabilities attribute, 54-8
CMIS security, 54-13
common types, 54-14
content management, 26-2
examples, 54-17, 54-22, 54-24, 54-27
Ext, 54-24
HTTP methods, 54-11
HTTP response status codes, 54-12
hypermedia, 54-11
hypermedia links, 54-7
hypermedia, navigating, 54-11, 54-17
link model, 54-4
Lists service, 48-4 to 48-34
People Connections service, 42-1 to 42-30
Activity Stream, 42-1
Connections, 42-10
Feedback, 42-23
Message Board, 42-26
Profile, 42-10
portal contact types, 54-15
projection parameter, 54-10
proxy server for, 54-16
q parameter, 54-10
query parameters, 54-10
rel attribute, 54-8
Resource Index, 54-5
resourceType attribute, 54-8
security, 54-13
security token parameter, 54-13
template attribute, 54-9
type attribute, 54-9
Rich Text Editor task flow
about, 31-3
parameters, 31-38
destinationID, 31-39
featuresOff, 31-39
initialSidebar, 31-39
Role Manager task flow, 69-16
round-trip development, 9-17
about, 1-8, 17-13
for WebCenter Portal: Spaces resources, 17-14
WebCenter Spaces, 56-12
RSS service
about, 53-1
accessing secure feeds, 53-6
adding task flow, 53-4
back-end requirements, 53-2
RSS Viewer task flow, 53-4
setting up a proxy server, 53-3
RSS Viewer task flow
about, 53-4
parameters
extAppId, 53-5
rssFeedLocation, 53-5
rssFeedLocation property, 53-5
runtime
portal, 9-19
runtime persistence, turning on and off, 64-32
runtime customizations, preserving, 9-19

S

sample application
features, 8-21
SampleWebCenterSpacesExtension.jws, 56-1
sandbox
about, 23-2
concurrency behavior, 23-6
destroying stale sandboxes, 23-7
disabling, 23-7
enabling, 23-3
saving customizations, 23-5
sandbox-namespaces, B-39
Scenario Editor, 67-3

creating a new scenario, 67-10
node types, 67-12
scenarios
file-based scenarios, 67-23
schemas
activities, 3-6
discussions, 3-6
installing, 3-6
installing with RCU (Repository Creation Utility), 3-7
installing with SQL script, 3-7
portlets, 3-6
WebCenter Portal, 3-6
scope property, 37-5, 45-13
scopeId property, 40-10
scopeName property, 49-13
scopename property, 19-9
scopes, 19-1
scripts
People Connections service
settings-insert.sql, 39-3
settings-update.sql, 39-3
scripts, deploy and configure, 10-12
Search - Saved Searches task flow, 45-9
search adapters, 45-22
attributes, 45-28
registering, 45-28
Search Metrics task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
Search Preferences task flow, 45-9
Search service
about, 45-1
adapters, 45-22
adding a search source, 45-22
adding to Framework applications, 45-9
APIs, 45-17
connection, 45-5
creating a Sitemap, 14-22
Oracle SES adapter, 45-15
security, 45-13
task flows
Saved Searches, 45-9
Search, 45-9
Search Preferences, 45-9
Search Toolbar, 45-9
troubleshooting, 45-33
Search task flow
about, 45-9
parameters
customAttributes, 45-11
hideRefiners, 45-11
keywordsInputRendered, 45-11
mimeType, 45-12
resourceId, 45-12
resourceScope, 45-12
resourceType, 45-13
searchScope, 45-12
serviceIds, 45-12
Search Toolbar task flow
about, 45-9
parameters
customAttributes, 45-11
hideRefiners, 45-11
mimeType, 45-12
scope, 45-13
searchBoxSize, 45-12
searchScope, 45-12
serviceIds, 45-12
searchBoxSize property, 45-12
searchScope property, 45-12
sectionWiseEditEnabled property, 40-10
security
about, 2-6
adding test users, 39-2
Composer, 18-50
configuring basic authentication, 69-26
customizable components, 69-39
Documents service, 30-2
EL expressions relating to, G-2
external applications, 69-33
identity propagation mechanisms, 69-44
JSR 286 portlets, 61-36
Notifications and, 49-13
overriding inherited security, 69-39
overview, 69-2
PDK-Java portlets, 69-49
People Connections service and, 39-2
portal pages, 9-9
portlets, 64-21, 69-39
registering custom certificates, 69-36
securing Analytics reports, 47-15
securing Recent Activities service, 50-7
securing RSS service, 53-6
subscriptions and, 49-13
WSRP producers, 69-45
WS-Security, 69-44
Security - Enterprise Role - Members Search task flow, 69-17
Security - Enterprise Role - Members Viewer task flow, 69-17
Security - Enterprise Role Members task flow, 69-17
security files, A-20
jazn-data.xml, A-20
jazn-data.xml sample, A-20
security policies, migration of, 10-24
security token parameter (REST APIs), 54-13
self-registration
building a custom self-registration page, 69-25
creating a self-registration page, 69-24
creating an invitation Page, 69-25
self-registration task flow, 69-24
self-registration task flow, 69-24
service IDs, G-13
serviceCategories property, 40-10

service-definition.xml file, 7-14, 7-17
serviceId property, 40-10, 44-8, 46-17
serviceIds property, 45-12
services
EL expressions relating to, G-12
See WebCenter Portal services
servlets
JSF servlet settings, 1-9
servlet mapping, 1-9
session storage
about, 61-55
checking for valid session, 61-57
enabling in provider.xml, 61-58
implementing in PDK-Java portlets, 61-56
session-options-factory element, B-39
setCustomAttribute API, 57-14
setGroupSpacesWebServiceEndpoint API, 57-10
settings-insert.sql script, 39-3
settings-update.sql script, 39-3
shortDesc portlet tag attribute, 64-27
Show Detail Frame
about, 18-49
action-level security, 18-54
adding to a page, 20-9
additional actions facet example, 22-2
applying action-level restrictions, 24-16
custom actions example, 22-14
disabling Edit option, B-23
enabling custom actions, 22-9
facets, 22-1
properties, B-16
style selectors, B-58
show modes
See portlet modes
Show Property component
about, B-26
ShowDetailsPage.jsp file, A-8
showDocuments property, 31-27, 31-32
showFolders property, 31-27, 31-29, 31-34
showIcons property, 14-6, 14-8
showInEditMode property, 37-5
showMinimizeAction portlet tag attribute, 64-28
showMoveAction portlet tag attribute, 64-28
ShowPage.jsp file, A-8
showRecursiveForums property, 34-10
showRemoveAction portlet tag attribute, 64-28
showResizer portlet tag attribute, 64-28
showRoot property, 14-5, 14-9
showUserDataOnly property, 37-5
Siebel applications, 55-2
Similar Items task flow
about, 46-7, 46-13
adding to a page, 46-14
default recipe, 46-13
editing properties, 46-15
parameters
classURN, 46-16
classURNRestrictions, 46-17
excludeObjectActions, 46-17
objectName, 46-16
recipe, 46-17
resourceId, 46-16
resourceType, 46-17
serviceId, 46-17
suppressPopup, 46-17
Similar Spaces task flow
about, 46-7, 46-12
adding to a page, 46-14
default recipe, 46-12
editing properties, 46-15
parameters
excludeObjectActions, 46-17
objectName, 46-16
recipe, 46-17
resourceId, 46-16
similarity calculations (Activity Graph)
about, 46-5
defaults for WebCenter Portal, 46-5
extending, 46-28
Simple Object Access Protocol (SOAP), 58-18
single sign-on
PDK-Java portlets, 69-50
Web Clipping, 58-7
site administrator design considerations, 2-14
Sitemap, creating, 14-22
skins
about, 1-8, 9-4, 15-1
applying
at design time, 15-11
at runtime, 15-12
attributes, 15-7
conditionally changing using EL expressions, 15-15
creating
as a portal resource, 15-8
at runtime, 15-11
CSS file, 15-3
for Spaces, 15-12
defining skin selectors, 15-4
deploying as shared library, 15-14
editing portal-skin.css, 15-5
exporting a skin, 15-10
files
adf-config.xml file, 15-17
trinidad-config.xml, 15-18
trinidad-skins.xml, 15-18
global style selectors, B-53
importing a skin, 15-9
Layout Customizable style selectors, B-56
making a JDeveloper skin available at runtime, 15-10
Page Customizable style selectors, B-54
Panel Customizable style selectors, B-56
portal, 9-16
portal resource
deleting a skin, 15-8
path for storing CSS file, 15-2
path for storing images, 15-3
updating a skin, 15-8
portlets style selectors, B-58

runtime administration
about, 15-6
enabling, 15-8
Show Detail Frame style selectors, B-58
skin compression, 15-17
skin coordination, 15-13
skin sharing, 15-13
style attributes, B-62
troubleshooting, 15-16
SOAP, 58-18
social networking, 38-1 to 38-12
Sortable Table View display template, 31-25
source control systems, Subversion, 4-1
sources folder, 9-18
spaceName property, 40-10
spaces property, 40-11
staging environment, 10-8
staging environment tasks, 10-5
staging environment, provisioning, 10-9
standalone Oracle WebLogic Server
deploying portlet applications, 62-11
testing portlet applications, 62-11
startFolderPath property
Document Explorer task flow, 31-27
Document List Viewer task flow, 31-29
Document Manager task flow, 31-31
Document Navigator task flow, 31-33
Folder Viewer task flow, 31-35
startNode property, 14-5, 14-7, 14-9
stateless resource proxying, 61-35
statusUpdateAllowed property, 40-11
Struts portlets
creating, 61-61
Oracle WebCenter Portal: Framework integration, 61-62
style classes for navigation task flow, 14-9
style property, 14-6, 14-8, 14-9
submitUrlParameters portlet tag attribute, 64-29
Subscription Preferences task flow
about, 49-3
parameters
hideActions, 49-13
scopeName, 49-13
Subscription Viewer task flow, about, 49-3
Subscriptions and Notifications, 49-1 to 49-20
about, 49-1
activity triggers, 49-1
adding namespace, 49-4
at runtime, 49-3
configuration file, 49-2
configuring, 49-4 to 49-14
connecting to BPEL server, 49-6
connecting to mail server, 49-5
delivery platforms, 49-2
effects of configurations
on Connections, 49-10
on Feedback, 49-11
on Message Board, 49-10
Java APIs, 49-14
messaging channels, 49-1
participant services and objects, 49-1
requirements, 49-2
setting application defaults, 49-7
task flows
adding to a page, 49-12
input parameters, 49-13
Subscription Preferences, 49-3, 49-11
Subscription Viewer, 49-3, 49-11
subscriptions and notifications
setting security, 49-13
using data controls, 49-16
subStyle property, 14-8
Subversion repository, creating, 4-2
suppressPopup property, 46-17

T

Tabbed View display template, 31-25
tabularView property, 36-13
Tag Center, about, 44-1
tag clouds, about, 44-2
Tagging - Dialog task flow
parameters
redirectURL, 44-8
Tagging - Dialog task flow, about, 44-4
Tagging - Personal View task flow
about, 44-4
serviceId parameter, 44-8
Tagging - Related Links task flow
about, 44-4
parameters
resourceId, 44-8
serviceId, 44-8
Tagging - Tag Cloud task flow
about, 44-4
resourceScope parameter, 44-8
Tagging - Tag Selection task flow
about, 44-4
resourceScope parameter, 44-8
Tagging - Tagged Items task flow
about, 44-5
resourceScope parameter, 44-8
tags
exposing resources to be tagged, 44-20
resource viewer, 44-20
Tags pane, 31-37, 31-39
Tags service
about, 44-1
adding tags through page templates, 44-16
APIs, 44-9
connection, 44-3
requirements, 44-3
Resource Action Handling framework, 44-20
security, 44-2, 44-8
Tagging Button component, 44-4
Tagging Menu Item component, 44-4
task flows
Dialog, 44-4
Personal View, 44-4
Related Links, 44-4

Tag Cloud, 44-4
Tag Selection, 44-4
Tagged Items, 44-5
troubleshooting, 44-21
task flows
Activity Graph service
Recommended Connections, 46-7, 46-11
Similar Items, 46-7, 46-13
Similar Spaces, 46-7, 46-12
adding to portal navigation, 11-9
Analytics service
Blog Metrics, 47-2
Discussion Metrics, 47-2
Document Metrics, 47-2
Login Traffic, 47-2
Page Traffic, 47-2
Portlet Instance Response Time, 47-2
Portlet Instance Traffic, 47-2
Portlet Response Time, 47-2
Portlet Traffic, 47-2
Search Metrics, 47-2
WebCenter Portal Traffic, 47-2
Wiki Metrics, 47-2
Announcements service
Announcements, 33-8
Announcements - Quick View task flow, 33-8
blogs
Blog Archives, 32-6
Blog Banner, 32-6
Blog Digest, 32-6
Blogs, 32-7
parameters, 32-12
Recent Blog Posts, 32-6
Content Presenter, 31-1
creating, 19-1
creating portlets from task flows, 59-5
Discussions service
Discussion Forums, 34-7
Discussions - Popular Topics, 34-7
Discussions - Quick View, 34-7
Discussions - Recent Topics, 34-7
Discussions - Watched Forums, 34-7
Discussions - Watched Topics, 34-7
Documents service, 31-2
AutoVue, 31-4
Document Explorer, 31-3
Document List Viewer, 31-3
Document Manager, 31-3
Document Mini Properties, 31-3
Document Navigator, 31-3
Document Properties, 31-3
Document Upload, 31-3
Document Version History, 31-4
Document Viewer, 31-3
Folder Viewer, 31-3
Recent Documents, 31-3
Rich Text Editor, 31-3
editing task flow content in Composer, 18-41
enabling custom actions, 22-7, 22-16
Events service
Calendar Main View, 52-7
Calendar Mini View, 52-7
external applications, 69-32
files affected, 4-10
guidelines for creating, D-1
List of Resource Types, 17-6
Lists, 48-2
Mail service
Mail, 36-11
navigation
Navigation - Breadcrumbs, 14-2
Navigation - Menu, 14-2
Navigation - Tree, 14-2
Notifications
Subscription Preferences, 49-3, 49-11
Subscription Viewer, 49-3, 49-11
Page service, Page - Create New task flow, 19-2
People Connections service
Activity Stream, 38-4
Activity Stream - Quick View, 38-4
adding to a page, 39-1
Connections, 38-5
Connections - Card, 38-5
Connections - Quick View, 38-6
Feedback, 38-6
Feedback - Quick View, 38-7
Message Board, 38-7
Message Board - Quick View, 38-8
Profile, 38-9
Profile - Snapshot, 38-9
Profile Gallery, 38-10
Publisher, 38-11
Polls service
Polls Manager, 37-3
Quick Poll, 37-3
Take Polls, 37-3
View Poll Results, 37-3
portlet entry, 59-7
properties
See properties
Recent Activities service, Recent Activities task flow, 50-3
RSS service, RSS Viewer task flow, 53-4
Search service
Saved Searches, 45-9
Search, 45-9
Search Preferences, 45-9
Search Toolbar, 45-9
Tags service
Dialog, 44-4
Personal View, 44-4
Related Links, 44-4
Tag Cloud, 44-4
Tag Selection, 44-4
Tagged Items, 44-5
taskFlowInstId property, 31-24, 31-29
teams
developer actions affecting metadata files, 4-4
developing in, 4-1
environment, 1-3

implementing common requirements, 4-12
managing files, 4-4
technology scopes
about, 6-1
adding, 6-2
default, 6-4
portal-specific, 6-5
removing, 6-2
template attribute, 54-9
templateCategory property, 31-24
templatefile property, 19-9
templates
Content Presenter
multiple content item display templates, 31-25
single content item display templates, 31-24
Content Presenter task flow, 29-2
page, 9-5
Portlet Producer application, 5-12
WebCenter Portal applications, 5-2
templateView property, 31-24
test environment, 10-6
title portlet tag attribute, 64-24
title property, 32-12
toDate property, 40-11
treeNavCollapsed property, 31-27
trinidad-config.xml, H-4
truncateAt property, 33-10
type attribute, 54-9

U

uitype property, 19-10
UPLOAD_MAX_DISK_SPACE parameter, 30-5
UPLOAD_MAX_MEMORY parameter, 30-5
UPLOAD_TEMP_DIR parameter, 30-5
uploadDocumentOnly property, 40-11
uploading
files to content repositories, 30-4
WebCenter Spaces resources, 56-12
URLs
adding to portal navigation, 11-9
in forms, 61-46
parameters, 61-43
rewriting for WSRP resource proxy, 61-35
stateless resource proxying, 61-35
UrlUtils.constructLink API, 61-45
use case examples
additional actions facet on Show Detail Frame, 22-2
enabling custom actions on a task flow, 22-14
user customization, 18-2
user Profile, EL expressions relating to, G-14
userId property, 40-11
userName property, 40-11
users
adding test users, 39-2
design considerations, 2-12
session, 61-55
usesUpload attribute on form, 38-10, 39-2
UTF-8 character sets, PDK-Java portlets, 60-17
utility, Preference Store Migration/Upgrade, E-24

V

validation-based caching, 58-21
about, 61-32, 61-59
in JSR 286 portlets, 61-33
in PDK-Java portlets, 61-61
value portable contact type, 54-16
Version History task flow
See Document Version History task flow
Versioning Navigator, 4-2
viaUser property, 40-11
View mode, 18-3
visibleAnnouncements property, 33-10
visibleForumsFetchSize property, 34-10
visibleTopicsFetchSize property, 34-10

W

WAR files, 6-6
Watched Forums task flow
parameters
doNotShowMoreLink, 34-10
visibleForumsFetchSize, 34-10
Watched Topics task flow
parameters
doNotShowMoreLink, 34-10
visibleTopicsFetchSize, 34-10
Web archive
See WAR
Web Clipping
about, 58-7, 66-1
adding certificates for trusted sites, E-18
adding to a page, 66-3
advanced features, 66-6
authenticating to external applications, 58-29
authentication, 58-7
browser support, 66-9
caching style, 58-22
capturing content, 58-24
charting, 58-26
comparing portlet building technologies, 58-13
configuring Oracle Advanced Security Option (ASO), E-19
configuring proxy settings, E-17
configuring repository, E-10
configuring security, E-18
creation style, 58-23
design time flexibility, 58-24
development tools, 58-22
expertise required, 58-17
fuzzy matching, 58-7
hiding and showing, 58-28
inline rendering, 58-7
intended users, 58-8
limitations, 66-8
multilingual support, 58-28
navigation, 58-7
open transport API, 58-7

pagination support, 58-28
parameter support, 58-27
personalization, 58-7
proxy authentication, 58-7
rendering inline, 58-25
resource tunneling, 58-7
restricting clipping from unauthorized external web sites, E-18
reusing web content, 58-7
rewriting image links, 66-8
single sign-on, 58-7
URL rewriting, 58-25
usage suitability, 58-16
use case, 58-8
using cascading style sheets (CSS), 66-8
Web Clipping Transport API, 66-7
Web Clipping portlet, 3-4
Web Clipping repository
configuring, E-10, E-14
creating database schema, E-13
JNDI data source, E-13
provider.xml path, E-11
repositoryInfo tag, E-15
using MDS, E-12
using Oracle database, E-13
Web Clipping Transport API, 66-7
Web Services for Remote Portlets
See WSRP
WebCenter Content content, propagation of content, 10-23
WebCenter Content Server
CMIS provider, 67-5
Personalization, 67-5
WebCenter Personalization
EL expressions, G-17
WebCenter Portal
Framework application
ADF security, 69-3
application roles, 69-2
automated security grants, 69-8
security, 69-2
Framework applications
application-level deployment profile, 70-11
deploying, 70-1
deployment profile, 70-10, 70-11
EAR file, 70-11
Enterprise Archive (EAR) file, 70-10
testing, 70-1
WebCenter Portal applications
ADF applications and, 6-5
creating, 5-1, 5-2
creating application manually, 6-5
creating pages, 5-8
creating through WAR files, 6-6
design considerations, 2-12
framework, overview, 2-2
life cycle, see life cycle, portal, 10-1
nightly builds, 10-5
planning, 2-1
propagation, 10-18
technology scopes, 6-4
template for, 5-2
testing, 10-5
WebCenter Portal extension bundle, installing, 3-1
WebCenter Portal Extension, about, 2-10
WebCenter Portal extension, installing, 3-1
WebCenter Portal services
about, 2-7
ADF security, 7-6
APIs, 7-2
data controls, 7-5
database connection, 7-9
EL expressions relating to
Documents service, G-14
People Connections service, G-14
Personalization service, G-17
services generally, G-12
external application connection, 7-12
files, A-23
adf-config.xml for the Documents service, A-24
adf-config.xml for the Worklist service, A-24
SES Web service connection example, A-23
general settings, 7-21
integrating, 2-9, 7-2
Lists service, 48-1 to 48-34
Recent Activities service, 50-2
RSS service, 53-1
service IDs, G-13
technologies, 7-1
Worklist service, 51-1
WebCenter Portal Traffic task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
WebCenter Portal: Spaces
about, 57-1
adding portlets, 57-3
APIs
about, 57-4
addMember, 57-16
case studies, 57-5, 57-8
createGroupSpace, 57-14
createGroupSpaceTemplate, 57-15
deleteGroupSpace, 57-15
getDiscoverableGroupSpaces, 57-20
getGroupSpaceMetadata, 57-20
getGroupSpaces, 57-19
getGroupSpaceTemplateMetadata, 57-22
getGroupSpaceTemplates, 57-22
getGroupSpaceURL, 57-24
getPublicGroupSpaces, 57-19
getRoles, 57-18
getWebCenterSpacesURL, 57-23, 57-25
GroupSpaceWSClient class, 57-12
GroupSpaceWSContext class, 57-11

GroupSpaceWSException class, 57-26
GroupSpaceWSMembers class, 57-16, 57-17
GroupSpaceWSMetadata class, 57-20, 57-23
inviteMember, 57-17
Javadoc, 57-27
removeMember, 57-17
security, 57-11
setCustomAttribute, 57-14
setGroupSpacesWebServiceEndpoint, 57-10
setting up WebCenter Portal applications, 57-8
exposing other applications, 57-53
integrating in Framework applications, 57-4
registering portlet producers, 57-3
See WebCenter Portal: Spaces
WebCenter Services
examples in the sample application, 8-21
WebCenter services
Announcements service, 33-1
Discussions service, 34-1
Documents service, 30-1
horizontal services, 7-1
IMP service, 35-1
Links service, 43-1
Mail service, 36-1
People Connections service, 38-1 to 38-12
about, 38-1 to 38-12
Activity Stream, 38-2, 38-3, 38-4
configuring, 39-1
Connections, 38-5
database set-up, 39-1
Feedback, 38-6
Message Board, 38-7
Profile, 38-9
Publisher, 38-11
requirements, 38-12
Polls service, 37-1
Search service, 45-1
Tags service, 44-1
WebCenter Services Portlets
about, 64-39
consuming
in Oracle Portal, 64-40
in Oracle WebCenter Interaction, 64-40
in Oracle WebLogic Portal, 64-40
portlet parameters
Activity Stream portlet, 64-45
Announcements portlet, 64-45
Blogs portlet, 64-44
common, 64-42
Discussion Forums portlet, 64-43
Document Manager portlet, 64-42
Mail portlet, 64-45
Polls Manager portlet, 64-44
Take Polls portlet, 64-45
portlets, 64-39
WebCenter Spaces
deploying custom shared libraries, 56-1
developing resources, 56-1
WebCenterComposerFilter, 23-14, B-48
WebLogic Managed Server
creating, 70-12
creating a connection, 70-13
provisioning, 70-12
WebLogic Server connection, 70-13
weblogic.xml, H-3
web.xml, 1-9, 1-10, B-47, H-5
width portlet tag attribute, 64-24
Wiki Metrics task flow
about, 47-2
adding to a page, 47-9
parameters
analyticsReportTitle, 47-14
analyticsResourceId, 47-14
applicationName, 47-14
maxDataPointsPerSeries, 47-14
wikis
about, 32-1
displaying links in Content Presenter, 32-5
enabling wiki functionality, 32-2
in Document Explorer task flow, 32-2
requirements, 32-1
window.location, iframes, 64-34
Worklist service
about, 51-1
add the Worklist service, 51-8
connections setup, 51-5
prerequisites, 51-2
security setup, 51-8
wsdl subdirectory, A-15
WSRP
about, 60-1
and JSR 286, 60-2
architecture, 60-2
Oracle WebCenter Portal: Framework architecture, 60-3
WSRP producers
about, 58-19
consumer preference store settings, E-22
database preference store settings, E-22
deleting portlets, 64-38
deregistering, 64-16
editing registration values, 64-15
external applications, 64-9
file-based preference store settings, E-22
mapping J2EE security roles, 64-10
producer test page, 64-5
refreshing, 64-16
registering, 64-3
security, 69-45
security attribute configuration, 64-6
supported versions, 64-3
syntax for WSDL URL, 64-4
testing connection, 64-16
WSRP test page, hiding or removing, E-20

Y

year property, 32-13